

THÈSE

PRÉSENTÉE

A L'UNIVERSITÉ PAUL SABATIER DE TOULOUSE

EN VUE DE L'OBTENTION

DU GRADE DE DOCTEUR D'ÉTAT (Sciences)

Spécialité : AUTOMATIQUE

PAR

Jean-Pierre RICHARD

Docteur-Ingénieur en Automatique

GÉNÉRATION DES TESTS DE DÉTECTION POUR LES SYSTÈMES LOGIQUES

Soutenue le 20 Décembre 1974 devant la Commission d'Examen :

MM. J. LAGASSE

Président

C. DURANTE

A. COSTES

R. G. BENNETTS

M. RAMSAY

P. AZEMA

} Examineurs



UNIVERSITE PAUL SABATIER DE TOULOUSE

DIRECTION DE L'UNIVERSITE

M. LARENG	Président
M. MATHIS	1er Vice-Président
M. MARTIN	2ème Vice-Président

U. E. R. SCIENTIFIQUES

HONORARIAT

M. BEDOS	Professeur honoraire
M. BLAIZOT	Doyen honoraire
M. CAPDECOMME	Doyen honoraire, Recteur honoraire, Correspondant de l'Institut
M. DUPOUY	Membre de l'Institut, Doyen honoraire, Directeur Général honoraire du C. N. R. S. Professeur honoraire.
M. DURAND Emile	Doyen honoraire
M. GAUSSEN	Professeur honoraire, Correspondant de l'Institut
M. JACQUES	Professeur honoraire
M. MASDUPUY	Professeur honoraire
M. MATHIS	Doyen honoraire
M. MIGNONAC	Professeur honoraire
M. MORQUER	Professeur honoraire, Correspondant de l'Institut
M. SECONDAT	Professeur honoraire
M. TEISSIE - SOLIER	Professeur honoraire
M. VANDEL	Professeur honoraire, Membre de l'Institut

PROFESSEURS

M. CASTERAS	Géologie, Correspondant de l'Institut
M. CAPDECOMME	Minéralogie, Correspondant de l'Institut
M. NOUVEL	Biologie Générale
M. GALLAIS	Chimie, Correspondant de l'Institut

M. DURAND Emile	Physique
M. FERT	Physique
M. LESBRE	Chimie Organique
M. BRUNEL	Physiologie Végétale
M. HURON	Mathématiques Appliquées
M. MARGULIS	Pédologie
M. TRICHE	Chimie Analytique et Spectrographie
M. LEDOUX	Zoologie Appliquée
M. RIVALS	Agriculture
M. PERRIER	Physique
M. MATHIS	Chimie
M. ORLIAC	Géochimie et Minéralogie
M. LAFOURCADE	Physique
M. ANGELIER	Zoologie
Melle de FERRE	Botanique
M. FARRAN	Minéralogie et Géotechnique
M. LAUDET	Physique Théorique et Calcul Numérique
M. SERFATY	Biologie Animale
M. LAGASSE	Electrotechnique
M. BLANC	Physique Nucléaire
M. LEREDDE	Botanique
M. LELUBRE	Géologie
M. TROCHAIN	Botanique
M. LALAGUE	Mathématiques Générales
M. BOUIGUE	Astronomie
M. COUCHET	Physique
M. LASCOMBES	Botanique
M. ASSELINEAU	Chimie Biologique
M. MAURET	Chimie Systématique
M. AGID	Physiologie
M. MONTANT	Cryptogamie
M. GAUTIER	Physique
M. CRUMEYROLLE	Mathématiques
M. GOURINARD	Géologie
M. PULOU	Minéralogie
M. CAMBOU	Physique Spatiale
M. MIQUEL	Chimie
M. LACOSTE	Electrotechnique
M. THIBAUT	Mécanique Rationnelle et Appliquée
M. MASCART	Mathématiques
M. MEDIONI	Psychophysiologie
M. RAYNAUD Pierre	Physiologie animale
M. ZALTA	Chimie Biologique
M. SEVELY	Electrotechnique
M. POMMIEZ	Mathématiques
M. REY Paul	Biologie Végétale
M. COULOMB	Physique
M. TRINQUIER	Physique
M. MARONI	Chimie

M. BEETSCHEN	Biologie Générale
M. DERACHE	Physiologie Animale
M. SATGE	Chimie Organique
M. LATTES	Chimie
M. VEDRENNE	Géophysique
M. DURAND-DELGA	Géologie
M. CARRARA	Physique
M. JEANNIN	Chimie Minérale
M. MAHENC	Chimie
M. MIROUSE	Géologie
M. BITSCH	Zoologie

PROFESSEURS ASSOCIES

M. ANDERSON

PROFESSEURS SANS CHAIRE

Mme BRUNEL	Physiologie Végétale
M. PICCA	Physique
M. MERIC	Mathématiques Appliquées
Mme LECAL	Zoologie
M. PILOD	Physique
M. LARROQUE	Physique
M. BARRANS	Chimie
Mme LAFON	Mathématiques
Melle LAPEYRE	Mathématiques-Informatique
M. BERTRAND	Chimie
M. DESQ	Mathématiques
M. ROCARD	Electronique
M. DEGEILH	Physique
M. POILBLANC	Chimie
M. TOUZE	Physiologie Végétale
M. FRASNAY	Mathématiques
M. CASSAGNAU	Zoologie
M. GUERIN	Mathématiques
M. PESCIA	Physique
M. SOUQUET	Géologie
M. SCHNEIDER	Biologie Cellulaire
M. SICARD	Biologie Génétique
M. de LOTH	Chimie Physique
M. MARTIN J.C.	Génie Electrique
M. SAPORTE	Physique
M. PERENNOU	Informatique
M. THENOZ	Génie Civil

M. REY Gérard	Electronique, Electrotechnique, Automatique
M. CAUSSINUS	Mathématiques Appliquées
M. DURAND Ph.	Physique
M. CASTAN	Informatique
M. REME	Mesures Physiques

MAITRES DE CONFERENCES

M. FONTAN	Physique Nucléaire
M. BAUDRAS	Chimie Biologique
M. ATTEIA	Mathématiques
M. CALVET	Mécanique des Fluides
M. PAGANI	Physique
M. CUPPENS	Mathématiques
Melle BARBANCE	Mathématiques
M. VERDIER	Physique
M. GILLY	Génie Mécanique
M. BEAUFILS	Informatique
M. COTTU	Génie Mécanique
M. ROUTIE	Génie Chimique
M. MARAL	Physique
M. BERTHELEMY	Zoologie
M. HURAUX	Physique
M. LEGRAND	Génie Civil
M. ABATUT	Electronique, Electrotechnique, Automatique
M. SOTIROPOULOS	Chimie Organique
Mme GERVAIS	Chimie Inorganique
M. BONEL	Chimie
M. MAUSS	Mécanique
M. TERJANIAN	Mathématiques

CHARGES D'ENSEIGNEMENT

M. PRADINES	Mathématiques
M. NGUYEN THANH VAN	Mathématiques
M. BANCEL	Mathématiques
M. BETOURNE	Informatique
M. CAMPAN	Psychophysiologie
M. CLERC	Mathématiques
M. GRIFONE	Mathématiques

INSTITUT NATIONAL POLYTECHNIQUE DE TOULOUSE

PROFESSEURS

M. NOUGARO	Hydraulique Générale et Appliquée
M. GARDY	Génie Chimique
M. BIREBENT	Electronique Appliquée
M. VOIGT	Chimie Minérale
M. MONTEL	Chimie Physique et Electrochimie
M. THIRRIOT	Hydrodynamique (E.N.S.E.E.I.H.T.)
M. GRUAT	Mécanique des Fluides (Hydraulique)
M. BUGAREL	Génie Chimique

PROFESSEURS SANS CHAIRE

M. HAMANT	Botanique
Melle BERDUCOU	Physiologie Végétale
M. DAT	Mécanique des Fluides
M. FALLOT	Biologie Végétale (Viticulture et Arboriculture)
M. TRUCHASSON	Mécanique
M. HOFFMANN	Electronique
M. ENJALBERT	Chimie

MAITRES DE CONFERENCES

M. MARTY	Physique Industrielle
M. LEFEUVRE	Electronique (E.N.S.E.E.I.H.T.)
M. ANGELINO	Génie Chimique
M. DABOSI	E.N.S.C.T.
M. MATHIEU	Chimie Analytique
M. TRANNOY	Physique (E.N.S.E.E.I.H.T.)
M. CASTAGNETTO	Mathématiques (E.N.S.E.E.I.H.T.)
M. GILOT	Génie Chimique (Institut de Génie Chimique)
M. CALMON	Techniques Agricoles
M. MATHEAU	Electronique

INSTITUT NATIONAL DES SCIENCES APPLIQUEES DE TOULOUSE

PROFESSEURS

M. ROQUES	Chimie Industrielle
M. GRATELOUP	Génie Electrique
M. MASO	Génie Civil
M. MIRA	Génie Electrique

PROFESSEURS SANS CHAIRE

M. COLETTE	Physique
M. FAGET	Physique
M. DURAND Gilbert	Chimie

MAITRES DE CONFERENCES

M. SCHUTTLER	Physique
M. LETURCQ	Génie Electrique
M. SIRIEYS	Mécanique et Génie Civil
M. BROUSSEAU	Physique

CHARGES D'ENSEIGNEMENT

M. LACAZE	Mathématiques
-----------	---------------

ADMINISTRATION

M. PRINEAU	Secrétaire Général de l'Université Paul Sabatier
------------	---

AVANT - PROPOS

Le travail présenté dans ce mémoire a été effectué au sein de l'Equipe "Structures Logiques Nouvelles" du LABORATOIRE d'AUTOMATIQUE et d'Analyse des Systèmes du Centre National de la Recherche Scientifique, dirigé par Monsieur le Professeur J. LAGASSE.

Nous tenons, tout d'abord, à exprimer notre profonde gratitude à Monsieur le Professeur J. LAGASSE pour la confiance qu'il nous a témoignée en nous accueillant au sein du Laboratoire et lui exprimer nos très vifs remerciements pour l'honneur qu'il nous fait en présidant notre Jury de thèse.

Monsieur C. DURANTE, Professeur à L'Université des Sciences et Techniques de Montpellier, a été notre premier guide en matière de recherche. Nous apprécions sa participation à ce Jury et l'assurons de notre profonde reconnaissance.

Monsieur A. COSTES, Maître de Conférences à l'Institut National Polytechnique de Toulouse, a bien voulu accepter de participer à cette Commission. Nous le remercions sincèrement pour l'attention avec laquelle il a examiné le manuscrit.

Monsieur M. RAMSAY, Chef du Groupe Informatique à la Direction des Recherches et Moyens d'Essais, a bien voulu faire partie de notre Jury de thèse malgré ses nombreuses charges. Nous lui en sommes très reconnaissants.

Que Monsieur R.G. BENNETTS, de l'Université de Southampton (G.B.), trouve ici l'expression de notre gratitude pour sa présence à cette Commission et l'intérêt qu'il a bien voulu accorder à nos travaux.

Chef de l'Equipe "Structures Logiques Nouvelles", Monsieur P. AZEMA, Maître de Recherche au C.N.R.S., nous a accordé sa confiance quand nous avons abordé ce thème de recherche. Nous lui savons gré des conseils et des encouragements qu'il a su nous prodiguer pendant la durée de cette étude.

Nous tenons à remercier les membres de l'Equipe "Structures Logiques Nouvelles" pour les nombreuses discussions critiques qui nous ont permis d'améliorer ce travail.

Nous ne saurions clore cet avant-propos, sans exprimer nos très vifs remerciements à tous ceux qui ont participé à la réalisation matérielle de ce mémoire.

INTRODUCTION

La vérification d'un système logique même de taille modérée est nécessaire à tous les stades de développement, fabrication et maintenance de ce système. La détection permet de vérifier si le comportement du système est correct ou non, alors que le diagnostic conduit à localiser une panne lorsqu'elle existe.

Ce mémoire est une étude des procédures de vérification à appliquer à un système logique pour effectuer la détection des pannes au cours de la maintenance.

Or, il est impossible de vérifier tous les cas de fonctionnement sauf pour les systèmes triviaux. Il s'agit donc de rechercher un sous-ensemble des configurations d'entrée, ordonné ou non, qui appliqué sur les entrées primaires permet, par simple observation des sorties, de détecter les pannes du système.

Un modèle de fonctionnement permet de décrire le comportement du système étudié. Deux types de modèle de fonctionnement sont généralement utilisés :

- d'une part, les structures de réalisation données sous forme d'interconnexion d'éléments physiques caractérisés par des entrées, des sorties et une correspondance entrées-sorties. Ce sont souvent les logigrammes.

- d'autre part, les modèles fonctionnels décrivent le comportement sans aucune référence à la réalisation. Ils sont généralement des tables, des graphes de fluence ou des fonctions booléennes associés au circuit.

Parce qu'il est nécessaire de choisir un sous-ensemble de toutes les possibilités d'entrée, on doit associer au modèle de fonctionnement, un modèle de panne. La définition de ce modèle dépend de deux paramètres qui sont la probabilité d'apparition de la panne et la simplicité de traitement. Il faut bien avouer, qu'en pratique, c'est cette deuxième condition qui est prépondérante. Ainsi pour les modèles en logigrammes ou sous forme de fonctions booléennes, on choisit en général les collages permanents d'une ou de plusieurs des connexions décrites ; pour les tables ou graphes de fluence les pannes sont permanentes et ne doivent pas augmenter le nombre des états de la machine.

Les méthodes de recherche des tests sont bien sûr liées au choix des modèles de fonctionnement et de panne. Dans les solutions classiques à partir d'un graphe ou d'une table de fluence, il s'agit plutôt d'une identification de la machine correcte parmi les machines en panne possibles. Tandis qu'avec un logigramme ou une fonction booléenne, on recherche successivement, pour chaque panne, à distinguer le modèle correct du modèle en panne.

Pour traiter le cas des systèmes séquentiels décrits par une structure de réalisation, ou par l'ensemble des fonctions booléennes associées aux variables internes et aux sorties, on dresse un modèle itératif dont chaque cellule représente un intervalle de temps pour le système séquentiel et l'on effectue une génération de tests de la même façon que pour un système combinatoire, en imposant à la séquence trouvée d'être applicable à partir de n'importe quel état initial.

L'efficacité des méthodes de génération s'appuyant sur un modèle de réalisation est généralement reconnue par rapport aux autres méthodes qui utilisent les fonctions booléennes ou les graphes de fluence lorsque l'objectif est l'obtention d'un ensemble réduit de tests. Cela provient, en particulier, du fait que, contrairement aux méthodes purement fonctionnelles, elles prennent en compte la spécificité de la réalisation.

La diversité des technologies de réalisation et leur évolution implique l'existence de nombreux modèles de fonctionnement du type structure de réalisation. A chacune de ces descriptions de comportement est attaché un modèle de panne qui est en général le plus probable ou du moins qui implique un ensemble de pannes fortement probable. Il en découle donc de nombreuses méthodes d'obtention des tests sur ces différentes structures de réalisation.

Il sera donc toujours nécessaire, dans un souci d'efficacité, de concevoir et de mettre en oeuvre des méthodes spécifiques des deux paramètres : description du fonctionnement et modèle de panne. Surtout si l'on désire traiter une classe de panne suffisamment large.

Les difficultés rencontrées lorsqu'on essaye d'une part de ramener tous les circuits à une même description, et d'autre part de transposer les modèles de panne d'une description à l'autre, montrent en effet que cette démarche présente un intérêt très limité.

Un modèle de fonctionnement est une approximation de la réalité. Par exemple, les modèles de réalisation ne tiennent pas compte des différents retards des éléments du circuit au niveau de la génération des tests. Cette simplification conduit à des résultats parfaitement valables lorsqu'il s'agit de circuits purement combinatoires, car dans ce cas les retards ne jouent pas un rôle prépondérant dans l'explication du comportement. Par contre, lorsqu'on traite de circuits séquentiels la présence de retards est essentielle. Cela conduit à remarquer que les séquences de tests obtenues ne sont en fait que des tests potentiels pour le circuit réel et qu'il est ensuite indispensable de les valider sur un modèle plus fin. Cette validation est le plus souvent réalisée par simulation du circuit comportant des retards, soit sur ordinateur, soit directement par simulation physique. On s'aperçoit alors que le pourcentage de tests potentiels rejetés lors de la validation est une fonction fortement croissante de la taille du circuit traité.

Cette remarque conduit donc à poser les limites de validité du modèle de fonctionnement. Le modèle de fonctionnement est valable avec certitude à l'intérieur du fonctionnement normal.

Ainsi, nous dirons que toute méthode de génération de tests, pour les systèmes séquentiels, sur un modèle de comportement qui ne contient pas la notion de retard, ne peut-être actuellement qu'une méthode heuristique.

Une solution à cette difficulté serait bien sûr de rechercher des méthodes de vérification permettant de prendre en compte, au niveau de la génération, les retards associés aux éléments de la structure traitée. Une telle approche est faite pour un type de circuit particulier dans (BRE 1). Malheureusement, on ne connaît pas de méthode d'obtention des tests qui ne soit, au moins en partie, une énumération des cas possibles. Ainsi, si l'on chiffre la complexité des méthodes en temps de calcul nécessaire, la recherche d'un test est un problème N-P complet (COC 1) qui nécessite, dans l'état actuel des connaissances, un temps de calcul croissant exponentiellement avec une mesure de la taille du circuit traité.

L'obtention des tests pour un système logique de grande dimension ne peut donc pas être envisagée à l'aide d'une telle solution qui amènerait évidemment une plus grande complexité des procédures. On est alors réduit à la génération de tests potentiels pour un modèle considéré sans retard puis à une vérification à posteriori des séquences obtenues.

La complexité du problème nous contraint à utiliser une description qui ne peut conduire qu'à des solutions heuristiques. Il semble donc hors de proportion de mettre en oeuvre des méthodes très sophistiquées qui ne seraient vraiment valables que pour le modèle.

Il est donc important de rechercher les conditions qui resteront valables dans la réalité et de les satisfaire en priorité. Ces conditions doivent être recherchées parmi celles que doit remplir une séquence pour être un test du modèle de fonctionnement.

Comme nous l'avons souligné précédemment le modèle est surtout représentatif du comportement lorsqu'il est utilisé pour déterminer les réponses à des séquences appartenant au fonctionnement normal, c'est-à-dire l'ensemble des séquences prévues lors de la conception du système.

Ainsi, on est conduit à une description fonctionnelle du comportement qui doit, pour aboutir à des solutions simples, à la fois être compréhensible et conserver un lien étroit avec la structure de réalisation. C'est seulement si elle s'appuie sur un langage évolué que la partie fonctionnelle d'une description peut permettre la compréhension humaine ; décrivant ainsi les commandes et les opérations logiques usuelles à l'aide de primitives de ce langage.

Il est d'ailleurs surprenant de constater que l'évolution des modèles de description n'ait pas été suivie par les chercheurs préoccupés de test des systèmes logiques alors que depuis 1950 (BAR 1) , une grande partie des recherches sur la conception s'est orientée vers la définition de langages élaborés permettant de pallier les défauts ou les insuffisances des modèles classiques. Cela ne s'explique même plus par le souci de rester proche de la réalisation physique puisque l'essor de la technologie vers une intégration toujours plus poussée rend, dans une certaine mesure, les schémas en portes élémentaires dépassés.

Les principaux résultats originaux que nous avons obtenus sont exposés en trois chapitres. Il s'agit de méthodes algébriques, de solutions directes sur des logigrammes et d'un nouveau modèle réaliste pour le traitement des systèmes logiques complexes. Il nous a semblé superflu de présenter à nouveau en détail les différentes approches classiques, d'une part, parce que les deux premiers chapitres recouvrent les principales approches pour la génération des tests par comparaison de deux modèles de comportement, d'autre part, parce que cette étude a déjà été menée à plusieurs reprises (COU 1) , (RIC 1) , (BEN 1).

Nous présenterons tout d'abord des méthodes pour le niveau circuit. Comme nous l'avons souligné, il s'agit de solutions spécifiques du modèle de fonctionnement et du modèle de panne choisis.

Au chapitre 1, nous traiterons deux méthodes algébriques d'obtention des tests de détection. Le modèle de fonctionnement utilisé est composé des fonctions booléennes associées à différents points du circuit traité. Le modèle de panne est du type collage permanent.

La première méthode s'appuie sur la notion de différence booléenne et permet de déterminer les tests pour les pannes uniques. Il est possible par cette approche d'aboutir à un ensemble optimal de tests.

La seconde méthode, par une analyse de l'influence des pannes sur l'expression de la fonction réalisée par un circuit, propose une solution pour le cas des pannes multiples sur les entrées primaires. Si l'on connaît de plus la structure du circuit, il est possible d'aboutir à un ensemble de tests pour les pannes multiples de toutes les connexions de ce circuit.

Le chapitre 2 présente deux méthodes utilisant directement le logigramme d'un circuit pour générer un ensemble de test de détection.

La première méthode est une transposition au niveau structure de la notion de différence booléenne. En approfondissant cette notion, on explique ainsi tous les algorithmes de propagation. Les temps de calcul obtenus s'avèrent en moyenne équivalents à ceux obtenus par le D-algorithme alors que la mise en oeuvre est beaucoup plus simple.

La seconde partie est une illustration d'une méthode adaptée à une réalisation. Elle traite d'un circuit cellulaire particulier qui permet de réaliser de manière économique toutes les fonctions combinatoires. Les notions de commande et d'observation conduisent à dégager une solution graphique simple pour traiter les pannes uniques aussi bien que les pannes multiples de cette structure.

Une critique du traitement des systèmes séquentiels, à l'aide du modèle itératif déduit de la structure, nous conduit alors à un nouveau modèle de description.

Le chapitre 3 expose ce nouveau modèle de description pour le test des systèmes logiques complexes. Il comprend deux parties : la première décrit le comportement du système à l'aide d'un organigramme dont les instructions sont écrites en langage A.P.L. La seconde est la structure de réalisation donnée sous forme d'interconnexion d'éléments. La correspondance entre les deux parties est assurée par une même dénomination des identificateurs. On possède ainsi une description qui permet à la fois la compréhension humaine et l'automatisation des tâches fastidieuses.

Nous avons ensuite défini les pannes de cette description comme un collage simple de la structure correspondant à une seule instruction. Un ensemble de conditions nécessaires est ensuite mis en évidence. Elles correspondent, en fait, aux notions classiques de commande-observation que l'on retrouve dans toutes les procédures de vérification. La détermination d'un ensemble de prétest, qui s'est avéré particulièrement efficace sur les exemples traités, est alors possible. Nous avons ensuite montré comment on pouvait utiliser ce modèle pour générer un test qui détecte une panne donnée. Il s'agit bien évidemment d'une procédure en grande partie à base d'énumération implicite.

1. MÉTHODES ALGÈBRIQUES

DE GÉNÉRATION DES TESTS

Nous présenterons dans ce chapitre deux méthodes algébriques d'obtention des tests de détection. Afin de permettre un traitement purement logique, nous supposerons que les pannes sont du type collage permanent des connexions à tester.

Le modèle de fonctionnement utilisé est donc composé de fonctions booléennes associées à certaines connexions du circuit. Bien évidemment, ces fonctions peuvent être déduites de la structure de réalisation, par exemple du logigramme de portes élémentaires ET, OU, PAS, NI, ON. Toutefois les temps de calcul et la place mémoire nécessaires à l'obtention de ces fonctions à partir de la réalisation montrent que ces procédures ne doivent être utilisées que dans les cas suivants :

- si l'on possède déjà les expressions de ces fonctions booléennes
- si l'on désire obtenir un ensemble optimal de tests pour les pannes uniques ou un ensemble de tests pour les pannes multiples.

Lorsqu'aucune de ces conditions n'est réalisée, il est alors préférable pour obtenir un ensemble de tests à partir de la structure de réalisation d'utiliser une méthode du type de celles que nous présenterons au chapitre 2.

La première méthode s'appuie sur la notion de différence booléenne qui permet théoriquement de traiter de façon similaire les collages simples et multiples. En fait, elle ne présente un réel intérêt pratique que pour les pannes simples. Nous ne présenterons donc que ce dernier cas. Le modèle de fonctionnement est composé des fonctions associées aux connexions à vérifier. Le modèle de panne est le collage à zéro ou à un d'une quelconque de ces connexions.

La seconde partie traite des pannes multiples pour un circuit logique. Suivant les informations que l'on possède sur le modèle de fonctionnement on peut obtenir un ensemble de tests plus ou moins efficace :

Si l'on connaît simplement la fonction réalisée par le circuit cela permet de générer les tests pour les pannes multiples des entrées primaires. Lorsqu'il s'agit d'un circuit en arbre ce même ensemble de tests détecte toutes les pannes multiples du réseau. Comme nous le verrons, les résultats obtenus dans ce cas théorique sont généralisables à un circuit quelconque. La procédure de génération des tests permet aussi de montrer que les points caractéristiques de première et de seconde espèce (faciles à obtenir à partir de l'expression de la fonction) constituent un ensemble de tests pour toutes les pannes multiples d'une réalisation monotone. Enfin, lorsque le circuit est quelconque, il est nécessaire de connaître le logigramme pour aboutir à un ensemble de tests pour toutes les pannes multiples.

1.1. METHODE DE DETECTION DES PANNES SIMPLES A PARTIR DE LA DIFFERENCE BOOLEENNE.

Comme pour toutes les méthodes de détection des pannes simples, il faut supposer que les circuits traités sont sans redondance (FRI 1).

Les hypothèses de collage permanent et de non redondance permettent d'affirmer que toute panne se traduit par un changement de l'image caractéristique de la fonction réalisée par le circuit.

Soit $Z = F(Y, x)$ une fonction booléenne générale des $n+1$ variables (Y, x) . Un collage de la variable x à la valeur booléenne α transforme la fonction Z en une nouvelle fonction $Z' = F(Y, \alpha)$. Considérons une combinaison $(Y_0, \bar{\alpha})$. Elle est un test pour la panne x collée à $\alpha(x/\alpha)$ si :

$$F(Y_0, \bar{\alpha}) \neq F(Y_0, \alpha)$$

ou encore si $F(Y_0, \bar{\alpha}) \oplus F(Y_0, \alpha) = 1$

Cette remarque conduit à introduire la notion de différence booléenne.

1.1.1. Définition de la différence booléenne

.Définition pour une fonction booléenne simple

Soit une fonction booléenne simple $f(Y,x)$ des $n+1$ variables (Y,x) . La différence booléenne par rapport à x est une fonction des n variables Y notée f_x et définie par la relation :

$$f_x = f(Y,x) \oplus f(Y,\bar{x})$$

$$\text{donc } f_x = f(Y,1) \oplus f(Y,0)$$

la fonction $f(Y,x)$ s'exprime à l'aide de la différence par rapport à x par

$$f(X,x) = f_x \cdot x \oplus f(Y,0).$$

.Définition pour une fonction booléenne générale

Soit $F(Y,x)$ une fonction booléenne générale à p composantes nous appellerons différence booléenne de F par rapport à x la fonction booléenne générale à p composantes et notée F_x obtenue en prenant la différence booléenne par rapport à x de chaque composante.

1.1.2. Condition nécessaire et suffisante

Considérons le circuit C figure 1.1. Il a n entrées $X = (x_1, x_2, \dots, x_n)$ et p sorties $Z = (z_1, \dots, z_p)$. Soit $Z = F(X) = (f_1, f_2, \dots, f_p)$ l'expression générale des p sorties en fonction des variables X . Si m est une connexion quelconque du circuit, Z peut se mettre sous la forme d'une fonction des $n+1$ variables (X,m) . Soit $Z = G(X,m) = (g^1, g^2, \dots, g^p)$ l'expression générale de la fonction Z et $m = m(X)$ une expression de la fonction attachée à la connexion m .

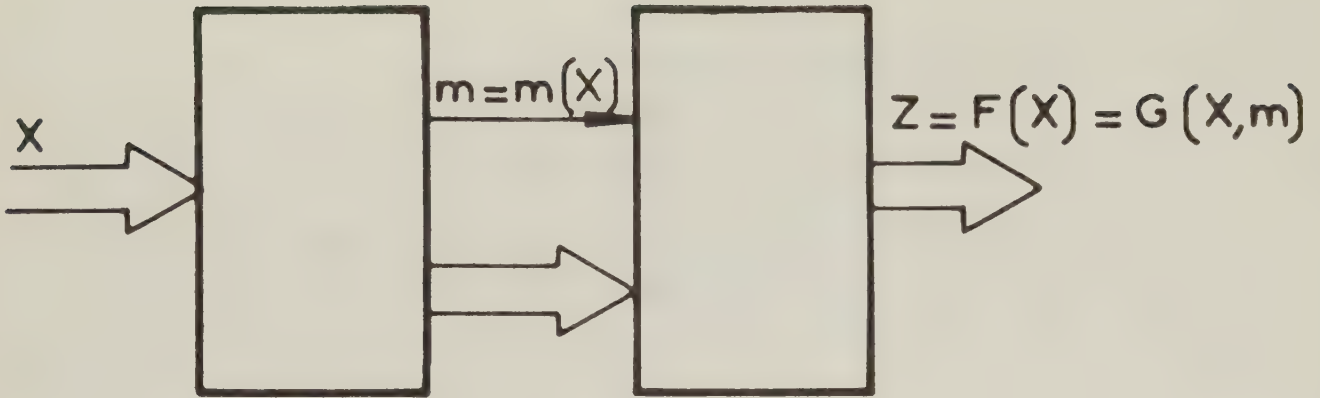


Figure 1.1. Test pour la connexion m

Théorème 1.1. Une condition nécessaire est suffisante pour qu'une combinaison X_0 des variables X soit un test pour la panne $m/\bar{\alpha}$ est qu'elle soit solution de l'équation booléenne $L(X) = 1$ avec

$$L(X) = G_m^+ \cdot m^\alpha = 1$$

où G_m^+ est le condensé additif de G_m (différence booléenne de G par rapport à m) et $m^\alpha = m$ si $\alpha = 1$ $\bar{m}$ si $\alpha = 0$).

Afin d'alléger la démonstration et sans que cela n'en diminue la généralité nous poserons $\alpha = 1$.

Démonstration :

On a $L(X_0) = G_m^+(X_0) \cdot m(X_0) = 1$ qui est logiquement équivalent à

$$\exists i \text{ tel que } \begin{cases} g_m^i(X_0) = 1 \\ m(X_0) = 1 \end{cases}$$

que l'on peut encore écrire en remplaçant g_m^i par son expression

$$\begin{cases} g^i(X_0, m(X_0)) \oplus g^i(X_0, \bar{m}(X_0)) = 1 \\ m(X_0) = 1 \end{cases}$$

Soit encore

$$\left\{ \begin{array}{l} g^i(X_0, m(X_0)) \neq g^i(X_0, 0) \\ m(X_0) = 1 \end{array} \right.$$

qui est équivalent à :

$$\left\{ \begin{array}{l} G(X_0, m(X_0)) \neq G(X_0, 0) \\ m(X_0) = 1 \end{array} \right.$$

ce qui démontre bien la propriété énoncée. Nous appellerons donc $L(X)$ fonction caractéristique de la panne $m/\bar{a}$. $\square$

1.1.3. Détermination d'un ensemble optimal de tests de détection

Soit un circuit combinatoire à n entrées primaires X et $M = \{ m \}$ un ensemble de p connexions de ce circuit pour lesquelles on désire déterminer un ensemble optimal de tests permettant de détecter les $2p$ collages de M . On connaît pour chacune des p connexions les deux fonctions associées $m(X)$ et $G(X, m)$.

On calcule donc pour toutes les connexions les deux fonctions caractéristiques associées et l'on obtient ensuite l'ensemble des tests désirés en déterminant un ensemble de combinaisons X_0 de cardinalité minimale et tel que chacune des $2p$ fonctions caractéristiques contienne au moins un des points de cet ensemble. La détermination de ces points s'effectue à l'aide d'une des méthodes classiques de résolution des tables de choix.

1.1.4. Détermination d'un ensemble sous-optimal de tests

Lorsqu'on ne recherche pas un ensemble optimal de détection, il existe deux variantes de cette méthode. La première consiste à supprimer la procédure de choix qui demande un temps de calcul important. Soit

$L_1, L_2, \dots, L_{2p}$ les $2p$ fonctions caractéristiques calculées. On effectue successivement le produit de L_1 avec chaque terme L_i de la liste $L_2, \dots, L_{2p}$. Si le produit $L_1.L_i$ est non nul on remplace L_1 par ce produit et l'on supprime L_i de la liste. A la fin de cette étape, on obtient dans L_1 le premier test. Puis l'on recommence la même procédure avec les termes de la liste non supprimés, cela jusqu'à épuisement de cette liste.

La seconde solution est surtout intéressante si l'on recherche les tests de détection pour les entrées primaires. En effet ainsi que l'a montré (BEN 1) il est possible à partir de l'expression de la fonction de calculer directement un seul point de la différence booléenne par rapport à l'une des variables. Ce qui bien sûr accélère encore notablement le temps de calcul.

1.1.5. Exemple d'application

L'exemple figure 1.2. est le schéma d'un circuit pour lequel on désire obtenir les tests pour les collages des entrées primaires $a, \dots, h$ et les collages de la connexion m .

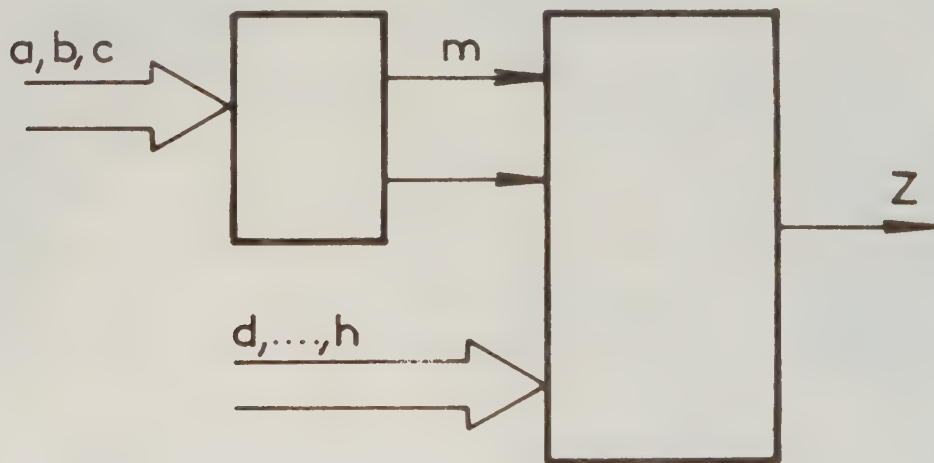


Figure 1.2. Exemple

$$\begin{aligned} Z &= ab\bar{d}\bar{e} + ac\bar{d}\bar{e} + fh\bar{d}\bar{e} + gh\bar{d}\bar{e} \\ &= (\bar{b}\bar{c}\bar{f}\bar{g} + \bar{d}\bar{e}).m + \bar{d}\bar{e}(f+g)h \\ m &= a(b+c) \end{aligned}$$

Le calcul des fonctions caractéristiques pour les collages des entrées et l'application de la procédure de groupement sous optimale fournissent l'ensemble des tests suivants :

a	b	c	d	e	f	g	h
0	1	X	0	0	1	X	1
1	X	1	0	0	1	X	0
0	X	X	0	0	0	0	1
1	0	0	0	0	0	X	0
X	0	0	1	X	1	X	1

A titre d'exemple, développons le calcul pour la connexion m.

$$Z_m = \bar{d}\bar{e}\bar{h} + \bar{d}\bar{e}\bar{f}\bar{g} + \bar{b}\bar{c}\bar{f}\bar{g}$$

des fonctions caractéristiques des collages à 0 et à 1 sont donc

$$L_0 = (\bar{d}\bar{e}\bar{h} + \bar{d}\bar{e}\bar{f}\bar{g} + \bar{b}\bar{c}\bar{f}\bar{g}) (b+c) a$$

$$L_1 = (\bar{d}\bar{e}\bar{h} + \bar{d}\bar{e}\bar{f}\bar{g} + \bar{b}\bar{c}\bar{f}\bar{g}) (\bar{b}\bar{c} + \bar{a})$$

On remarque que L_0 et L_1 contiennent tous deux un point de l'ensemble déjà trouvé et donc l'ensemble des cinq tests précédent est l'ensemble de détection cherché.

Cette méthode a été programmée (CHI 1) et permet d'obtenir ces résultats directement à partir d'un logigramme de portes élémentaires.

1.2. DETECTION DES PANNES MULTIPLES

L'intérêt pratique des méthodes algébriques pour le traitement des pannes uniques est, comme nous l'avons souligné, discutable dans certains cas. En particulier, lorsque l'on connaît la réalisation il est beaucoup plus rapide d'utiliser directement la structure du circuit pour générer un

ensemble de détection sous optimal. Cela est possible parce que le nombre de pannes uniques pour un circuit combinatoire reste assez faible : $2p$ si le circuit possède p connexions à vérifier.

Par contre, il semble à priori impossible de traiter les pannes multiples autrement que globalement. En effet, ce nombre de pannes est $3^p - 1$ pour un circuit de p connexions. En réalité, ce nombre diminue considérablement si l'on effectue tout d'abord un groupement des pannes en classes d'équivalence et si l'on supprime les pannes impliquées par d'autres (SCH 1), (VER 1). Le nombre de ces pannes reste toutefois trop élevé pour envisager leurs traitements successifs.

Ainsi quel que soit le modèle de fonctionnement que l'on possèdera au départ, on se ramènera à un modèle algébrique.

Nous présenterons tout d'abord (paragraphe 1.2.1.) la génération des tests pour les collages multiples des entrées primaires. Si on se limite à ces pannes, il suffit alors de connaître la fonction que réalise le circuit.

Lorsqu'on sait de plus que le circuit est arborescent ou est une réalisation monotone, toutes les pannes multiples des connexions du circuit sont aussi détectées par la même procédure (paragraphe 1.2.2. et 1.2.3.).

Par contre, pour un circuit combinatoire quelconque, cette méthode se généralise mais il est alors nécessaire de connaître sa structure de réalisation en portes ET, OU, PAS, NI, ON, (paragraphe 1.2.4.).

1.2.1. Détection des pannes multiples sur les entrées primaires

Définition 1.1. Un diviseur d'un monôme m , que nous noterons $d-m$, est un monôme obtenu en supprimant un nombre quelconque des lettres de m . Nous noterons d_1-m un diviseur de m par une seule de ses lettres.

Définition 1.2. m est un monôme premier de $f(X)$, si m est inclus dans $f(X)$ et que chacun de ses diviseurs contient au moins un point de $\bar{f}(X)$

$$m \text{ est premier} \Leftrightarrow \{ m \leq f(X) ; \forall d-m, d-m \cdot \bar{f}(X) \neq \emptyset \}$$

Considérons un circuit et la fonction qui lui est associée $f(X) = \sum_{i \in I} m_i$ écrite sous forme d'une base première. Soit $\bar{f}(X) = \sum_{j \in J} M_j$ une base première de $\bar{f}(X)$.

Une panne sur les entrées primaires se traduit par une transformation des fonctions $f(X)$ et $\bar{f}(X)$ en des fonctions en panne notées $P f(X)$ et $P \bar{f}(X)$.

Une expression de $P f(X)$ est

$$P f(X) = \sum_{i \in I_1} d-m_i + \sum_{i \in I_2} m_i + \sum_{i \in I_3} 0$$

avec $I = I_1 \cup I_2 \cup I_3$ où I_1 , I_2 et I_3 sont des ensembles d'indices disjoints définis par :

$$I_1 = \{ i_1 \mid m_{i_1} \text{ est transformé en } d-m_{i_1} \text{ par la panne} \}$$

$$I_2 = \{ i_2 \mid m_{i_2} \text{ n'est pas transformé par la panne} \}$$

$$I_3 = \{ i_3 \mid m_{i_3} \text{ est rendu identiquement nul par la panne.} \}$$

Théorème 1.2. On obtient un ensemble de tests de détection pour les pannes multiples des entrées primaires en choisissant un point dans chacun des ensembles définis par $d-m_i \cdot \bar{f}(X)$ et $d-M_j \cdot f(X)$ où m_i et M_j sont les implicants premiers de $f(X)$ et $\bar{f}(X)$ pour tout $i \in I$ et $j \in J$.

Démonstration :

Supposons que l'ensemble I_1 soit différent de zéro, c'est-à-dire que la panne se traduit par la division d'un implicant premier de $f(X)$ au moins. Soit i_1 un élément de I_1 . Par définition d'un monôme premier

(définition 1.2.) il existe toujours au moins un point appartenant à l'ensemble défini par l'expression $d-m_{i_1} \cdot \bar{f}(X)$. Ce point appartient à $\bar{f}(X)$. La sortie a donc la valeur zéro lorsque le circuit est sans panne. Par contre c'est aussi un point de $P-f(X)$ et le circuit fournit donc la valeur 1 si la panne est présente. Le théorème est donc démontré pour toute panne se manifestant par un ensemble I_1 non vide.

Supposons maintenant que l'ensemble I_1 soit vide. Pour qu'il y ait panne, il faut alors que l'ensemble I_3 ne soit pas vide. Cela n'est possible que s'il existe au moins un monôme M_j de $\bar{f}(X)$ transformé en l'un de ses diviseurs $d-M_j$. Cette panne est alors détectée par un point de $d-M_j.f(X)$.

Pour tout diviseur $d-m_i$ ($d-M_j$) obtenu en supprimant plus d'une lettre du monôme premier m_i (M_j) il existe au moins un diviseur par une seule lettre d_1-m_i (d_1-M_j) tel que $d_1-m_i \subset d-m_i$ ($d_1-M_j \subset d-M_j$). Donc en prenant un point de $d_1-m_i \cdot \bar{f}(X)$ ($d_1-M_j \cdot f(X)$) on est assuré de prendre aussi un point de $d-m_i \cdot f(X)$ ($d-M_j \cdot f(X)$). Ce qui démontre le théorème. $\square$

1.2.2. Détection des pannes multiples dans un circuit en arbre

Nous supposerons dans cette partie que les circuits considérés sont réalisés à l'aide de portes élémentaires ET, OU, PAS, NI, ON. Le cas des circuits en arbre est sans intérêt pratique mais comme nous le verrons (paragraphes 1.2.3. et 1.2.4.) les résultats obtenus se généralisent aux cas des réseaux monotones et des circuits quelconques.

Définition 1.3. Un circuit est en arbre si chaque porte n'a qu'une seule sortie et que chaque entrée primaire n'attaque qu'une seule porte.

Définition 1.4. Une fonction $f(X,a)$ est monotone croissante (décroissante) par rapport à a si $f(X,0) \leq f(X,1)$ ($f(X,0) \geq f(X,1)$). Une fonction monotone par rapport à chacune de ses variables est dite monotone.

Propriété 1.1. Lorsqu'un circuit est un arbre de portes ET, OU, PAS, NI, ON, la fonction réalisée est monotone. Sa base première est unique.

Afin de simplifier l'exposé nous ne parlerons que de fonctions monotones croissantes dont la base première complète ne contient que des variables sous la forme normale. Toutefois les résultats obtenus peuvent être transposés sans modification à toute fonction monotone.

Définitions 1.5. Considérons une fonction monotone croissante $f(X)$ et notons $\bar{f}(X)$ la fonction complémentaire. Soient $f(X) = \bigvee_{i \in I} m_i$ et $\bar{f}(X) = \bigvee_{j \in J} M_j$ les bases premières complètes correspondantes.

Chaque implicant premier m_i contient un point caractéristique de première espèce qui est obtenu en remplaçant chaque lettre présente dans m_i par 1 et chaque lettre absente par 0. Chaque implicant premier M_j contient un point caractéristique de seconde espèce obtenu en remplaçant chaque lettre présente par zéro et en affectant la valeur 1 à chaque variable absente.

Théorème 1.3. Les points caractéristiques de première et de seconde espèce constituent un ensemble complet de tests pour les pannes multiples d'un circuit arborescent.

Démonstration :

Toute panne sur les connexions de sortie d'une porte ET, OU, PAS, NI, ON est équivalente à une panne simple ou multiple sur les entrées de cette porte (SCH 1). En appliquant cette équivalence successivement, on montre donc que toute panne multiple est équivalente à une panne multiple sur les entrées primaires du circuit.

La méthode de génération des tests pour les pannes multiples des entrées primaires nous permet donc d'obtenir les tests pour les pannes multiples de tout le circuit. Puisque toutes les variables sont positives dans l'expression de $f(X)$ et négatives dans $\bar{f}(X)$ on peut choisir comme tests les points caractéristiques de première et seconde espèce de $f(X)$ □

1.2.3. Détection des pannes multiples dans une réalisation monotone

Soit un circuit composé de portes ET, OU, PAS, NI, ON.

Définition 1.6. Nous dirons qu'il s'agit d'une réalisation monotone si pour tout couple de points α et β du circuit et deux chemins quelconques joignant α et β les nombres d'inversions n_1 et n_2 le long de ces deux chemins vérifient la réalisation

$$n_1 \equiv n_2 \pmod{2}$$

Définition 1.7. Le développement en arbre d'un circuit quelconque est une transformation qui fait correspondre à un circuit quelconque C_I un circuit en arbre $C_A = \tau(C_I)$ obtenu en appliquant les règles suivantes :

- la sortie de C_A est la sortie du circuit initial. La construction est effectuée en se déplaçant de la sortie vers les entrées primaires ;
- toute porte de C_I ayant une sortance s est remplacée par s portes fonctionnellement identiques ayant le même nombre d'entrées et de sortance 1.
- toute entrée primaire "a" qui attaque p portes est remplacée par p entrées primaires "a", indicées de 1 à p .

Un exemple d'application de cette transformation est donné figure 1.3.a. et 1.3.b.

Propriété 1.2. Une réalisation monotone fournit une fonction monotone.

Dans la suite, nous ne considérerons que les fonctions monotones croissantes sans que cela ne diminue la généralité des résultats.

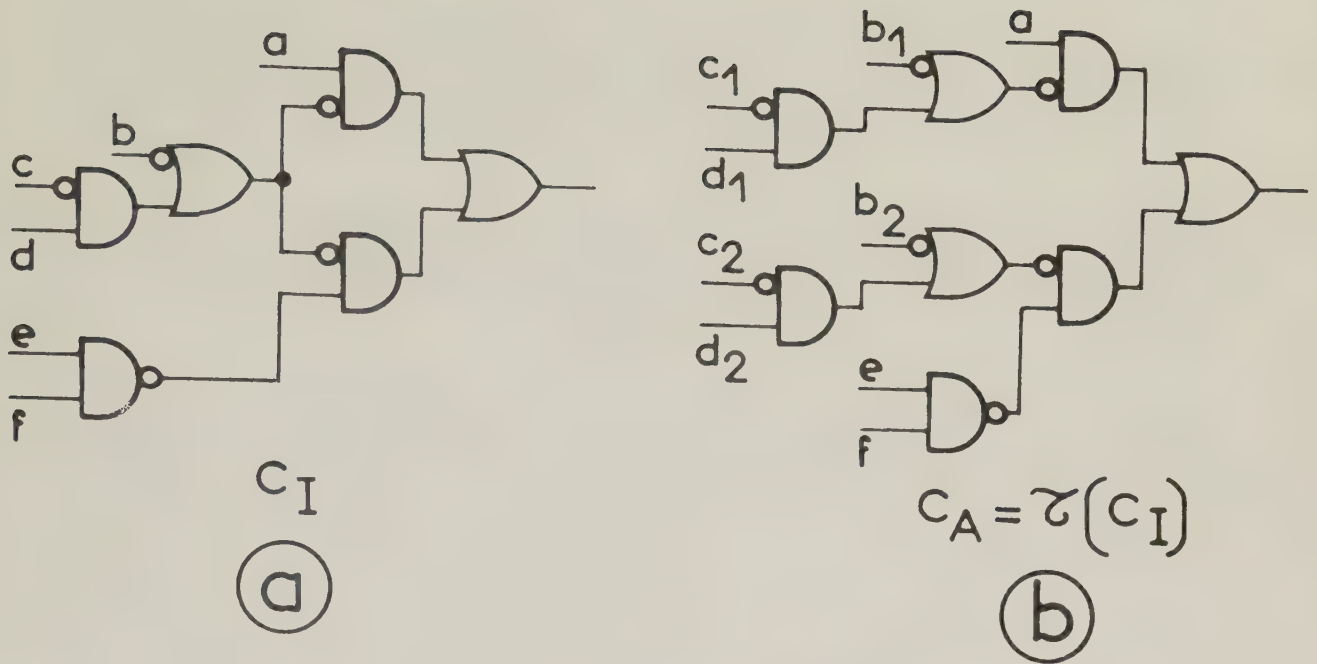


Figure 1.3.

Théorème 1.4. Les points caractéristiques de première et de seconde espèce constituent un ensemble complet de tests pour les pannes multiples d'une réalisation monotone.

Démonstration :

A tout circuit C_I la transformation τ fait correspondre un circuit en arbre C_A . Cette correspondance τ , induit de manière naturelle une application qui à toute panne simple ou multiple et à toute combinaison d'entrée du circuit initial fait correspondre respectivement une panne simple ou multiple et une combinaison d'entrée du circuit en arbre C_A . En particulier, si l'on considère une panne quelconque de C_I et un test pour cette panne, l'application τ leur fait correspondre une panne et un test pour cette panne de C_A . Réciproquement, une combinaison qui détecte une panne de C_A est l'image d'un test pour la panne correspondante de C_I , si ce test et cette panne existent pour C_I .

Pour calculer les tests de C_I on peut donc rechercher un ensemble de tests pour C_A qui soit applicable à C_I . Afin de démontrer le théorème 1.4. nous passerons par l'intermédiaire des deux lemmes 1.1. et 1.2.

Lemme 1.1. Les implicants premiers de la fonction $g(X_A)$ réalisée par C_A sont les images soit d'implicants premiers, soit de termes inclus dans un implicant premier de la fonction $f(X_I)$ réalisée par C_I .

En effet, on peut obtenir une expression de $f(X_I)$ en opérant une réduction des variables X_A sur la base première de $g(X_A)$. (L'opération de réduction consiste à rendre égales les variables de même nom et d'indices différents). Comme $g(X_A)$ et $f(X_I)$ sont toutes deux monotones croissantes, toutes les variables sont sous la forme normale dans l'expression de $f(X_I)$ obtenue par réduction. Il n'existe donc aucun consensus possible et la seule simplification éventuelle sera l'inclusion de première espèce. Ce qui démontre le lemme 1.1. $\square$

Cette propriété est bien sûr valable pour les expressions des fonctions complémentées $\bar{g}(X_A)$ et $\bar{f}(X_I)$.

Lemme 1.2. Parmi les implicants premiers de $g(X_A)$ ($\bar{g}(X_A)$), il existe au moins un terme image d'un implicant premier quelconque de $f(X_I)$ ($\bar{f}(X_I)$).

En effet, comme nous l'avons remarqué précédemment la seule simplification possible à partir d'une expression réduite de $g(X_A)$ ($\bar{g}(X_A)$) est l'inclusion de première espèce. Ce qui démontre bien que l'expression obtenue contient bien tous les implicants premiers de la fonction $f(X_I)$ ($\bar{f}(X_I)$). $\square$

Considérons donc un implicant de $g(X_A)$ soit m et calculons un de ses diviseurs $d-m$. Pour obtenir le test correspondant pour C_I , il faut alors choisir un point de l'ensemble $R(d-m, \bar{g}(X_A))$ où R est l'opération de réduction. Soit w le terme de $\bar{g}(X_A)$ tel que $R(d-m, w)$ différent de zéro. $R(w)$ est soit un implicant premier de $\bar{f}(X_I)$ et donc cet ensemble contient un point caractéristique soit un multiple d'un implicant premier

de $\bar{f}(X_I)$ alors il existe (lemmes 1.1. et 1.2.) w' tel que $R(w')$ est implicatif premier de $\bar{f}(X_I)$ et $R(w') \leq R(w)$. L'ensemble $R(d-m, w') = R(d-m) \cdot R(w')$ est différent de zéro et contient un point caractéristique. Ce qui complète la démonstration du théorème. $\square$

1.2.4. Détection des pannes multiples pour une réalisation quelconque

De même que dans le cas des pannes simples pour obtenir un ensemble de détection pour les pannes multiples d'un circuit qui n'est ni arborescent ni une réalisation monotone, la fonction associée n'est pas une information suffisante. Il est nécessaire de retenir la topologie du circuit à traiter.

La méthode de génération des tests est calquée sur la démonstration que nous avons utilisée pour le théorème 1.3.

Considérons un circuit C_I quelconque et soit $C_A = \tau(C_I)$ le circuit en arbre associé, $f(X_I)$ et $g(X_A)$ sont les fonctions réalisées par C_I et C_A . Nous avons déjà noté la possibilité d'obtenir une expression de $f(X_I)$ en opérant une réduction sur l'ensemble des variables X_A . Ces différentes transformations sont schématisées à l'aide d'un exemple sur la figure 1.4.a. pour un circuit C_I sans panne. La figure 1.4.b. reprend le même diagramme pour C'_I , circuit C_I affecté par la panne simple $\alpha/1$. On remarque sur cet exemple que $\alpha/1$ n'est équivalente à aucune panne sur les entrées primaires de C_I mais qu'elle est par contre équivalente à l'une des pannes $b_1/1$, $c_1/1$ sur le circuit C_A .

Ainsi le passage par C_A permet de ne considérer que des pannes sur les variables primaires de la fonction $g(X_A)$.

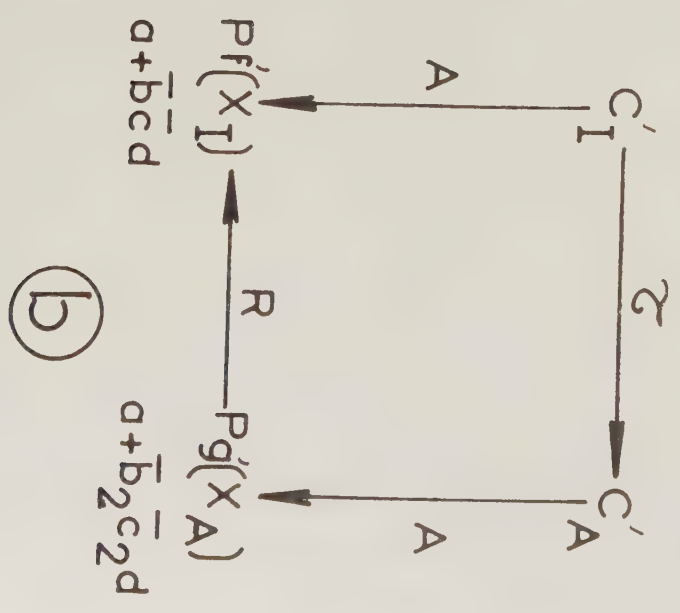
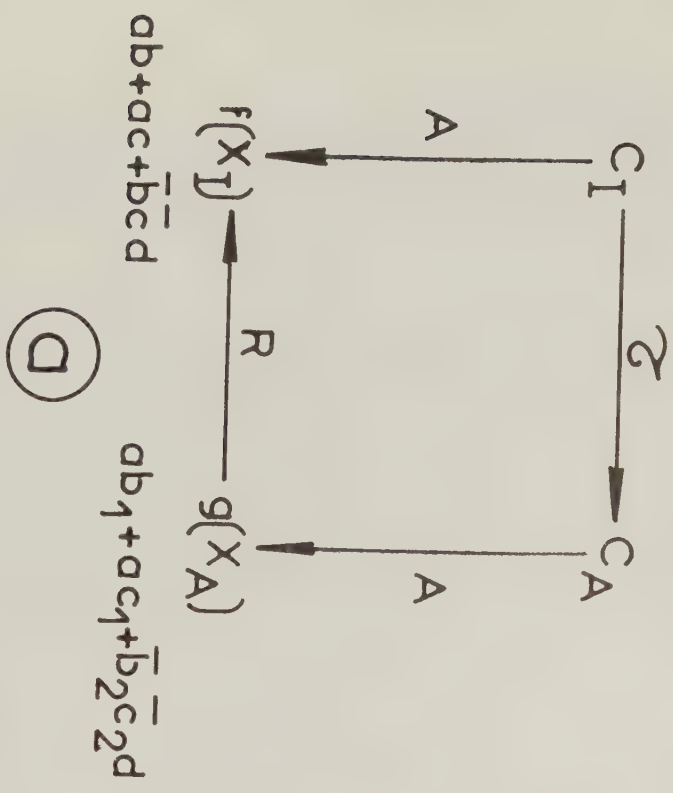
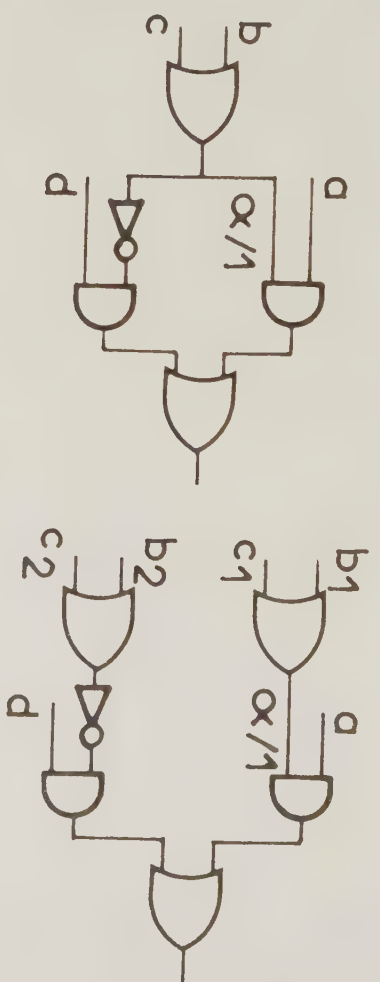
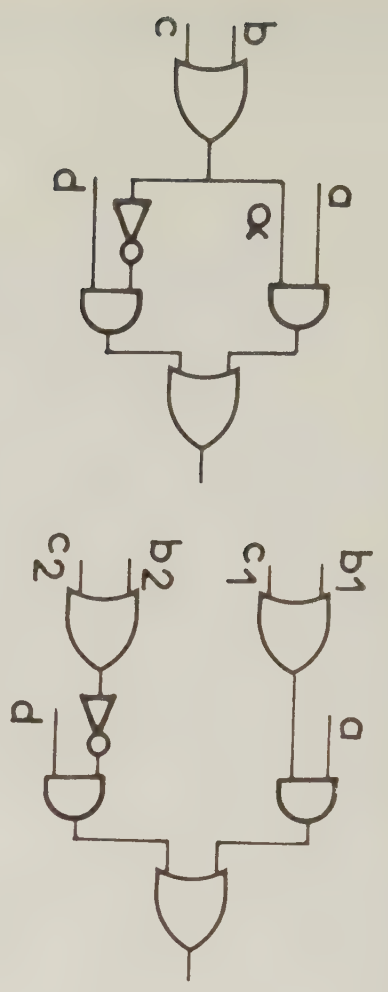


Figure 1.4.

- C_I : circuit initial
- C_A : circuit en arbre associé $C_A = \tau(C_I)$
- A : calcul de la fonction associée au circuit
- R : opération de réduction

Théorème 1.5. La génération des tests pour les pannes multiples d'un circuit quelconque peut être effectuée de la façon suivante :

1. A partir du circuit initial C_I , on construit le développement en arbre de ce circuit : C_A .
2. On calcule la fonction et la fonction complémentée associée à C_A : $g(X_A)$ et $\bar{g}(X_A)$.
3. Pour chaque diviseur d'un implicant m de $g(X_A)$ ($\bar{g}(X_A)$) on effectue le produit des différents diviseurs de m avec $\bar{g}(X_A)$ ($g(X_A)$).
4. On applique l'opération de réduction à chaque produit et on choisit un point par produit. L'ensemble de ces points est un ensemble de détection des pannes multiples.

Démonstration :

En effet, toute panne de C_I est équivalente à une panne sur les entrées primaires de C_A et l'on peut donc obtenir une expression de la fonction $P g(X_A)$ associée au circuit en panne C'_A en transformant l'expression $g(X_A)$. L'expression associée à C'_I -circuit initial en panne- est obtenue en appliquant l'opération de réduction à l'expression $P g(X_A)$; l'exemple de la figure 1.4.b. montre ces différentes étapes. Remarquons toutefois que dans ce cas, il n'est pas possible de calculer seulement les diviseurs par une seule lettre. $\square$

1.2.5. Définition d'une fonction test

La définition d'une fonction-test retenant les propriétés topologiques nécessaires à la détection des pannes multiples permet de réduire considérablement le volume de calcul. Cette fonction a le même nombre d'implicants premiers que $g(X_A)$ et le même nombre de variables que $f(X_I)$.

Définition 1.8. Un implicant compatible de $g(X_A)$ ou $\bar{g}(X_A)$ est transformé en un implicant non nul par l'opération de réduction R .

Définition 1.9. Un implicant incompatible de $g(X_A)$ ou $\bar{g}(X_A)$ est transformé en un implicant identiquement nul par l'opération de réduction R .

En effet, lorsque l'on calcule $g(X_A)$ ou $\bar{g}(X_A)$ on peut obtenir des termes contenant une même variable sous la forme normale et sous la forme complémentée (par exemple $a_i \bar{a}_j$).

Définition 1.10. L'opération de réduction modifiée $\hat{R}$ est définie sur l'ensemble des variables X_A :

- pour l'ensemble des implicants compatibles par $\hat{R} \equiv R$
- pour l'ensemble des implicants incompatibles l'opération de réduction modifiée sera identique à R sauf pour les variables biformes qui seront remplacées par la variable portant un accent circonflexe.

Exemple :

$$R(a_1 b_1 b_2 \bar{c}_3) = ab\bar{c}$$

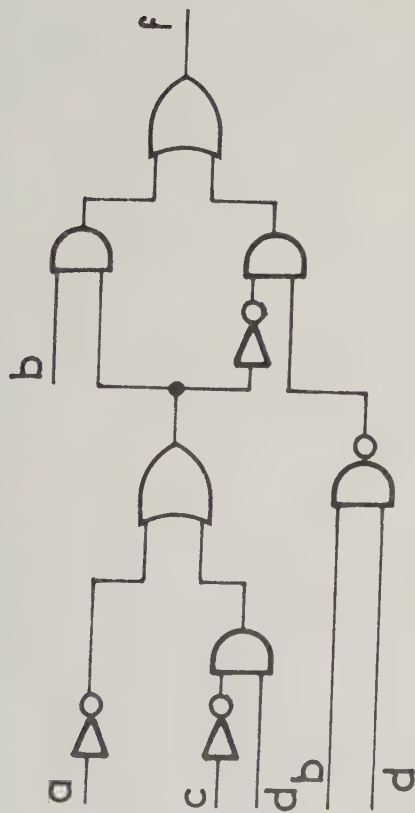
$$R(a_1 b_1 \bar{b}_2 \bar{c}_3) = 0$$

$$\hat{R}(a_1 b_1 b_2 \bar{c}_3) = R(a_1 b_1 b_2 \bar{c}_3)$$

$$\hat{R}(a_1 b_1 \bar{b}_2 \bar{c}_3) = a\hat{b}\bar{c}$$

Définition 1.11. Les fonctions-test sont obtenues à partir des expressions de $g(X_A)$ et $\bar{g}(X_A)$ en appliquant l'opération de réduction modifiée $\hat{R}$.

Un exemple de calcul des fonctions-test est donné figures 1.5.a. et b.

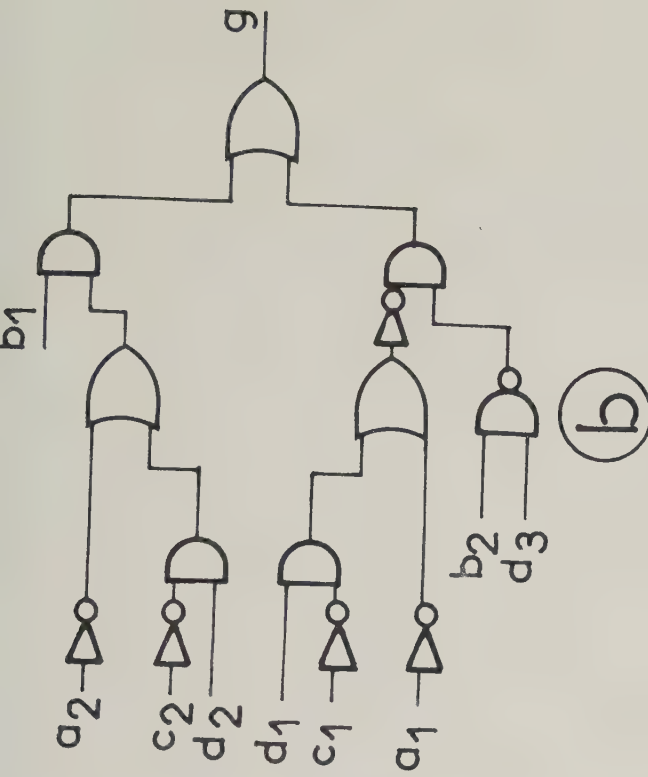


(a)

$$f = \bar{a}b + b\bar{c}d + a\bar{d} + abc$$

$$\bar{f} = \bar{a}\bar{b} + \bar{b}\bar{c}d + abcd$$

Figure 1.5.a.



(b)

$$g = \bar{a}_2b_1 + b_1\bar{c}_2d_2 + a_1\bar{b}_2c_1 + a_1\bar{b}_2\bar{d}_1 + a_1c_1\bar{d}_3 + a_1\bar{d}_1\bar{d}_3$$

$$\bar{g} = \bar{a}_1\bar{b}_1 + \bar{b}_1\bar{c}_1d_1 + a_2b_2c_2d_3 + \bar{a}_1a_2c_2 + a_2\bar{c}_1c_2d_1$$

$$+ \bar{b}_1b_2d_3 + a_2b_2\bar{d}_2d_3 + \bar{a}_1a_2\bar{d}_2 + a_2\bar{c}_1d_1\bar{d}_2$$

Figure 1.5.b.

$$\hat{R}(g) = \hat{f} = \bar{a}b + b\bar{c} + a\bar{b}c + a\bar{d} + b\bar{d}$$

$$\hat{R}(\bar{g}) = \hat{\bar{f}} = \bar{a}\bar{b} + \bar{b}\bar{c}d + abcd + a\bar{d} + \bar{b}d + \bar{a}c$$

$$+ a\bar{c}d + ab\bar{d} + a\bar{c}\bar{d}$$

Figure 1.5. Calcul des fonctions-test.

Définition 1.12. Nous appellerons maximums compatibles $CM(\hat{m})$ d'un incompatible $\hat{m}$ un groupe de monômes obtenus à partir de $\hat{m}$ en remplaçant les variables portant un accent par l'ensemble des combinaisons possibles de ces variables.

Exemple : $\hat{m} = a\bar{b}\bar{c}\hat{d}$

$$CM(\hat{m}) = \{a\bar{b}\bar{c}d, ab\bar{c}\bar{d}, a\bar{b}c\bar{d}, a\bar{b}\bar{c}\bar{d}\}$$

Montrons comment la procédure donnée par le théorème 1.5. se trouve transformée par l'utilisation des fonctions-tests $\hat{f}$ et $\hat{\bar{f}}$.

Théorème 1.6 . La génération des tests de détection pour les pannes multiples d'un circuit quelconque peut être effectuée à l'aide de la procédure suivante :

1. à partir du circuit initial C_I on construit le développement en arbre C_A de ce circuit.
2. on calcule les fonctions $g(X_A)$ et $\bar{g}(X_A)$ associées à C_A .
3. on applique l'opération de réduction modifiée $\hat{R}$ à $g(X_A)$ et $\bar{g}(X_A)$ qui fournit les fonctions tests $\hat{f}(X_I)$ et $\hat{\bar{f}}(X_I)$.
4. pour chaque terme des fonctions-tests :
 - a. s'il s'agit d'un implicant compatible, on calcule les diviseurs et l'on effectue le produit avec la fonction complémentaire ;
 - b. s'il s'agit d'un terme incompatible, on forme les maximums compatibles ; pour chacun d'eux, on calcule les diviseurs et les produits de ces diviseurs avec la fonction complémentaire.

Cette dernière procédure revient tout simplement à appliquer l'opération de réduction sur les variables X_A avant d'effectuer le produit avec la fonction complémentée.

Pour démontrer la validité de la procédure, analysons l'influence d'une panne sur un terme quelconque.

Soit M un implicant de $g(X_A)(\bar{g}(X_A))$ et posons $\hat{R}(M) = m$.
Considérons une panne qui transforme M en $d-M$.

1. M est un implicant compatible.

-a. $\hat{R}(d-M) = m$; cette transformation n'est pas à considérer comme une panne pour C_I puisqu'elle ne change pas l'implicant m .

exemple : $M = ab_1b_2c$ $d-M = ab_1c$

$$\hat{R}(M) = R(d-M) = abc = m$$

-b. $\hat{R}(d-M) = d-m$; dans ce cas il est possible de calculer l'effet de la panne directement comme un diviseur du terme m de la fonction-test.

exemple : $M = ab_1b_2c$ $d-M = b_1b_2c$

$$\hat{R}(d-M) = bc = d-m$$

2. M est un implicant incompatible : $R(M) = 0$

-a. $R(d-M) = 0$ comme dans le cas 1.a. il ne s'agit pas d'une panne pour C_I

exemple : $M = ab_1\bar{b}_2\bar{c}$ $R(M) = 0$

$$d-M = b_1\bar{b}_2\bar{c} \quad R(d-M) = 0$$

-b. $R(d-M) \neq 0$ dans ce cas $R(d-M)$ peut être obtenu directement comme un diviseur d'un des maximums compatibles de M .

exemple : $M = ab_1\bar{b}_2\bar{c}$ $R(M) = 0$

$$d-M = a\bar{b}_2\bar{c} \quad R(d-M) = a\bar{b}_2\bar{c} \neq 0$$

$$\hat{R}(M) = a\bar{b}_2\bar{c} \quad CM(a\bar{b}_2\bar{c}) = \{ ab_1\bar{b}_2\bar{c}, a\bar{b}_2\bar{c} \}.$$

L'étude exhaustive des différents cas possibles montre bien que l'on peut effectuer l'opération $\hat{R}$ avant le calcul des diviseurs. $\square$

On peut encore simplifier la procédure en remarquant qu'il n'est pas nécessaire de calculer tous les diviseurs d'un terme.

Théorème 1. 7. Lorsqu'un diviseur d-m d'un terme de la fonction-test a un produit non nul avec la fonction complémentaire, il n'est pas nécessaire de calculer les diviseurs de d-m.

En effet, soit m un terme de $\hat{f}(X_I)$ et d-m un de ses diviseurs. Supposons que le produit d-m . $\bar{f}(X_I) \neq 0$. La procédure implique le choix d'un point de ce produit, tout diviseur de d-m soit d(d-m) est tel que $d(d-m) \geq d-m$, on peut ainsi affirmer que $d(d-m) . \bar{f}(X_I) \geq d-m . \bar{f}(X_I)$. Donc le choix d'un point de d-m . $\bar{f}(X_I)$ est suffisant. Ce qui démontre bien le théorème. $\square$

1.2.6. Exemples d'applications

a. Tests pour les pannes multiples des entrées primaires

Reprenons le circuit donné figure 1.2. La fonction réalisée par ce circuit est :

$$z = abd\bar{e} + acd\bar{e} + fh\bar{d}\bar{e} + gh\bar{d}\bar{e}$$

$$\bar{z} = d + e + \bar{a}\bar{h} + \bar{b}\bar{c}\bar{h} + \bar{a}\bar{f}\bar{g} + \bar{b}\bar{c}\bar{f}\bar{g}$$

Ces deux expressions de z et $\bar{z}$ sont sous la forme de bases premières complètes. Il suffit donc de calculer les diviseurs par une seule lettre de chaque terme. Effectuons le calcul pour le premier terme de z soit $abd\bar{e}$, les diviseurs par une seule lettre sont :

$$b\bar{d}\bar{e}, a\bar{d}\bar{e}, ab\bar{e}, ab\bar{d}$$

En effectuant le produit avec $\bar{z}$ et en ne retenant que le premier terme non nul, on obtient :

$$\bar{a}\bar{h}\bar{b}\bar{d}\bar{e}, \bar{b}\bar{c}\bar{h}a\bar{d}\bar{e}, dab\bar{e}, eab\bar{d}$$

Remarque : Si l'on désire avoir un ensemble de tests aussi réduit que possible, on peut effectuer le produit d'un diviseur par la suite des termes déjà calculés puisque ces derniers sont inclus dans $\bar{z}$.

En utilisant cette remarque, on aboutit à l'ensemble des tests suivants :

$$\begin{array}{l}
 \text{produits dans } \bar{z} \left\{ \begin{array}{l} \bar{a} \, b \, c \, \bar{d} \, \bar{e} \, f \, g \, \bar{h} \\ a \, \bar{b} \, \bar{c} \, \bar{d} \, \bar{e} \, \bar{h} \\ a \, b \, c \, d \, \bar{e} \, f \, g \, h \\ a \, b \, c \, \bar{d} \, e \, f \, g \, h \\ \bar{a} \, \bar{d} \, \bar{e} \, \bar{f} \, \bar{g} \, h \end{array} \right. \\
 \\
 \text{produits dans } z \left\{ \begin{array}{l} a \, b \, \bar{c} \, \bar{d} \, \bar{e} \, \bar{f} \, \bar{g} \, \bar{h} \\ \bar{a} \, \bar{b} \, \bar{c} \, \bar{d} \, \bar{e} \, \bar{f} \, \bar{g} \, \bar{h} \\ a \, \bar{b} \, \bar{c} \, \bar{d} \, \bar{e} \, \bar{f} \, \bar{g} \, \bar{h} \\ \bar{a} \, \bar{b} \, \bar{c} \, \bar{d} \, \bar{e} \, \bar{f} \, g \, h \end{array} \right.
 \end{array}$$

b. Tests pour les pannes multiples d'un circuit en arbre

Remarquons que la fonction associée au circuit figure 1.2. est monotone. Ce circuit peut être réalisé sous forme d'un arbre. Dans ce cas, l'ensemble des tests obtenus précédemment détecte toutes les pannes multiples de cet arbre. Si l'on utilise le théorème 1.2. l'obtention des tests est encore simplifiée. Les points caractéristiques s'obtiennent en effet directement à partir des bases premières complètes de z et $\bar{z}$ (voir définitions 1.5.a. et b.). Il vient alors l'ensemble des tests suivants :

$$\begin{array}{l}
 \begin{array}{c} a \, b \, c \, d \, e \, f \, g \, h \\ a \, b \, \bar{d} \, \bar{e} \rightarrow 1 \, 1 \, 0 \, 0 \, 0 \, 0 \, 0 \, 0 \\ a \, c \, \bar{d} \, \bar{e} \rightarrow 1 \, 0 \, 1 \, 0 \, 0 \, 0 \, 0 \, 0 \\ f \, h \, \bar{d} \, \bar{e} \rightarrow 0 \, 0 \, 0 \, 0 \, 0 \, 1 \, 0 \, 1 \\ g \, h \, \bar{d} \, \bar{e} \rightarrow 0 \, 0 \, 0 \, 0 \, 0 \, 0 \, 1 \, 1 \end{array} \left. \vphantom{\begin{array}{c} a \, b \, c \, d \, e \, f \, g \, h \\ a \, b \, \bar{d} \, \bar{e} \rightarrow 1 \, 1 \, 0 \, 0 \, 0 \, 0 \, 0 \, 0 \\ a \, c \, \bar{d} \, \bar{e} \rightarrow 1 \, 0 \, 1 \, 0 \, 0 \, 0 \, 0 \, 0 \\ f \, h \, \bar{d} \, \bar{e} \rightarrow 0 \, 0 \, 0 \, 0 \, 0 \, 1 \, 0 \, 1 \\ g \, h \, \bar{d} \, \bar{e} \rightarrow 0 \, 0 \, 0 \, 0 \, 0 \, 0 \, 1 \, 1 \end{array}} \right\} \text{points caractéristiques de première espèce}
 \end{array}$$

	a	b	c	d	e	f	g	h	
d →	1	1	1	1	0	1	1	1	} points caractéristiques de seconde espèce
e →	1	1	1	0	1	1	1	1	
$\bar{a} \bar{h}$ →	0	1	1	0	0	1	1	0	
$\bar{b} \bar{c} \bar{h}$ →	1	0	0	0	0	1	1	0	
$\bar{a} \bar{f} \bar{g}$ →	0	1	1	0	0	0	0	1	
$\bar{b} \bar{c} \bar{f} \bar{g}$ →	1	0	0	0	0	0	0	1	

c. Tests pour une réalisation monotone

Le circuit figure 1.2. est réalisable par une structure monotone. Dans ce cas, le théorème 1.3. nous permet de dire que les points caractéristiques sont encore un ensemble de tests pour toutes les pannes multiples.

On peut toutefois ne pas utiliser cette remarque et générer les tests par la méthode générale donnée ci-dessous.

d. Tests pour les pannes multiples d'un circuit quelconque

Considérons le circuit de la figure 1.5. pour lequel nous avons déjà calculé les fonctions-tests.

$$\hat{f} = \bar{a}b + b\bar{c} + a\bar{b}c + a\bar{d} + b\bar{d}$$

$$\hat{\bar{f}} = \bar{a}\bar{b} + \bar{b}\bar{c}d + abcd + \hat{a}\bar{d} + \hat{b}d + \hat{a}c + a\hat{c}d + ab\hat{d} + a\hat{c}\hat{d}$$

pour les termes compatibles la procédure est identique à celle utilisée pour le circuit en arbre, on obtient les tests suivants :

$\bar{a} \bar{b} \bar{c} d$	} produits dans $\bar{f}$
$a b c d$	
$a \bar{b} \bar{c} d$	
$\bar{a} \bar{b} c$	

$\bar{a} \ b \ \bar{c} \ d$	} produits en f
$a \ \bar{b} \ \bar{c} \ \bar{d}$	
$a \ b \ c \ \bar{d}$	
$a \ b \ \bar{c} \ d$	
$\bar{a} \ b \ c \ d$	

Effectuons maintenant le traitement du terme incompatible $\bar{a}\bar{d}$. Les maximums compatibles correspondants sont $a\bar{d}$ et $\bar{a}d$, $a\bar{d}$ ne conduit à aucun autre test puisqu'il donne un produit non nul avec $a\bar{b}\bar{c}\bar{d}$. Par contre, $\bar{a}d$ donne un nouveau test qui est obtenu en effectuant $\bar{a}\bar{d} \cdot f = \bar{a}b\bar{d}$ qui doit donc être rajouté aux produits dans f.

1.2.7. Algorithmes mis en oeuvre

Afin de préciser sans ambiguïté la procédure générale pour obtenir les tests de détection d'un circuit quelconque, nous donnerons ci-dessous les algorithmes qui ont été programmés en Fortran IV. Comme la génération des tests pour les entrées primaires ou pour un circuit en arbre est incluse dans la procédure générale (pour le traitement des termes compatibles) nous décrirons tout d'abord cet algorithme.

Génération des tests pour un circuit en arbre

On suppose ici que la seule donnée est le logigramme du circuit.

1. Calcul de $f(X)$ et $\bar{f}(X)$ sous forme de bases premières complètes : $f(X) = \sum_{i \in I} m_i$, $\bar{f}(X) = \sum_{j \in J} m_j$

2. Traitement des termes de $f(X)$

2-1 $i = 1, k = 1$

2-2 $d = 1$

2-3 Calcul du diviseur de m_i par la dième lettre soit $d-m_i$

- 2-4 si $d-m_i$ est le premier diviseur calculé
 - aller en 2-5
 - sinon aller en 2-8
- 2-5 $j = 1$
- 2-6 Calcul de $P = d-m_i \cdot m_j$ (m_j implicant de $\bar{f}(X)$)
- 2-7 Si $P = 0$ faire $j = j+1$ et aller en 2-6
 - sinon faire $T(k) = P$, $k = k+1$, aller en 2-12
- 2-8 $l = 1$
- 2-9 Calcul de $P = (d-m_i) \cdot T(l)$
- 2-10 Si $P = 0$ aller en 2-11
 - sinon $T(l) = P$, aller en 2-12
- 2-11 Si $l = k$, aller en 2-5
 - sinon faire $l = l+1$, aller en 2-9
- 2-12 Si tous les diviseurs de m_i ont été calculés
 - aller en 2-13
 - sinon faire $d = d+1$, aller en 2-3
- 2-13 Si tous les implicants de $f(X)$ ont été considérés
 - aller en 2-14
 - sinon, faire $i = i+1$, aller en 2-2
- 2-14 Fin de traitement de $f(X)$.

3. Traitement de $\bar{f}(X)$

Cette partie est identique à la partie 2 en remplaçant f par $\bar{f}$.

La liste T contient les tests pour le circuit considéré.

Génération des tests de détection pour un circuit quelconque

- 1. Détermination du circuit en arbre C_A associé au circuit initial C_I .
- 2. Calcul des fonctions associées à C_A , $g(X_A)$, $\bar{g}(X_A)$

3. Calcul des fonctions-tests par application de l'opération de réduction modifiée : $\hat{R}$

$$\hat{f}(X_I) = \hat{R}(g(X_A)) ; \hat{\bar{f}} = \hat{R}(\bar{g}(X_A)).$$

4. Traitement des termes compatibles de $\hat{f}$ et $\hat{\bar{f}}$ étapes 2 et 3 de l'algorithme précédent.

5. Traitement des termes incompatibles

- 5-1 $ii = 1$
- 5-2 $jj = 1$
- 5-3 Former les compatibles maximums de $\hat{m}_{ii}$ soit CM_{jj}
- 5-4 $l = 1$
- 5-5 Calcul de $P = T(l).CM_{jj}$
- 5-6 Si $P = 0$ aller en 5-7
 - sinon $T(l) = P$, aller en 5-2
- 5-7 Si $l = k$, aller en 5-8
 - sinon $l = l+1$, aller en 5-5
- 5-8 $j = 1$
- 5-9 Calculer $P = CM_{jj} . m_j$
- 5-10 Si $P = 0$, aller en 5-11
 - sinon $T(k) = P$, $k = k+1$, aller en 5-3
- 5-11 Si $j = \text{fin}$, aller en 5-12
 - sinon $j = j+1$, aller en 5-9
- 5-12 Si $jj = \text{fin}$, aller en 5-13
 - sinon $jj = jj+1$, aller en 5-3
- 5-13 Si $ii = \text{fin}$, aller en 5-14
 - sinon $ii = ii+1$, aller en 5-2
- 5-14 Fin du traitement de $\hat{f}$.

6. Traitement de $\hat{\bar{f}}$ partie incompatible identique à la partie 5 en échangeant les rôles joués par $\hat{f}$ et $\hat{\bar{f}}$.

La liste T contient tous les tests.

1.3. CONCLUSIONS

L'intérêt essentiel des méthodes booléennes est qu'elles permettent d'exprimer globalement la ou les conditions que doit remplir une combinaison d'entrée pour être un test. Cela conduit à différentes solutions dont certaines sont indépendantes du type de réalisation des circuits traités. Leur généralité permet aussi d'admettre d'autres types de pannes que les collages uniques à zéro et à un des connexions et d'atteindre, lorsque cela s'avère nécessaire, un ensemble de tests de cardinalité minimale.

Malheureusement, le modèle de fonctionnement est composé des fonctions booléennes associées aux connexions à vérifier et cela constitue pratiquement un handicap sérieux. En effet, la recherche de ces fonctions booléennes à partir d'une structure de réalisation nécessite un temps de calcul important.

Le programme pour les pannes multiples écrit en Fortran IV traite sur IBM 360/75 un circuit de 50 portes élémentaires en environ 30 s. Mais il est très difficile d'évaluer de manière générale le temps de calcul nécessaire pour un circuit quelconque.

Alors que pour des circuits de 150 portes, le calcul s'est avéré possible, pour d'autres circuits de la même dimension, il a été abandonné car le nombre des termes de la fonction-test était trop grand.

En pratique, les méthodes booléennes seront limitées par le temps de calcul et la place mémoire à des circuits ne dépassant pas une centaine de portes élémentaires lorsque l'on désire un ensemble de tests pour toutes les connexions du réseau. Des circuits de taille plus importante pourront être traités pour obtenir uniquement les tests pour les entrées primaires car dans ce cas la fonction associée aux sorties du réseau est suffisante aussi bien pour les pannes simples que pour les collages multiples.

Devant cette difficulté, il est apparu indispensable de développer des méthodes s'appliquant à des circuits de taille plus importante avec un temps de calcul plus raisonnable. La rançon de ce gain de temps est la diminution de la généralité des procédures. Elles sont alors spécifiques de la réalisation et du modèle de panne. Elles conduisent à des ensembles de tests en général non optimaux. A titre d'exemple de ces méthodes, nous avons choisi de présenter au chapitre 2 des solutions s'appuyant sur le logigramme d'un circuit.

2. MÉTHODES D'OBTENTION DES
TESTS DE DÉTECTION SUR
UN LOGIGRAMME

Comme nous l'avons vu les méthodes algébriques sont intéressantes pour le traitement des pannes multiples. Pour les pannes uniques, elles permettent d'aboutir à un ensemble optimal de tests de détection en s'appuyant sur les fonctions booléennes associées aux connexions à vérifier. Enfin, certaines méthodes sont indépendantes de la réalisation et donc ne nécessitent pas de nouvel effort de mise en oeuvre lorsque l'on traite des circuits appartenant à des types de réalisation différents.

Malheureusement, on ne dispose pas toujours d'une expression des fonctions associées aux connexions et leur calcul s'avère très long. Aussi, lorsque l'on possède une structure de réalisation du circuit à vérifier et que l'on désire simplement obtenir un ensemble réduit de tests pour les pannes uniques, il est plus efficace de rechercher les combinaisons directement sur la structure.

Les structures de réalisation peuvent être très diverses et donc l'un des inconvénients majeurs des méthodes que nous présenterons dans ce chapitre est qu'elles sont spécifiques du modèle de fonctionnement que nous avons choisi. Ainsi, au niveau des circuits logiques élémentaires composant un système il sera toujours nécessaire, si l'on recherche l'efficacité de traitement, de concevoir des méthodes et des programmes spécifiques de la structure de réalisation.

Les solutions que nous exposerons dans ce chapitre sont donc proposées à titre d'exemples des principes que l'on peut mettre en oeuvre pour aboutir à un ensemble de détection des pannes uniques à l'aide de la structure de réalisation.

La première partie concerne une méthode de transformation du logigramme à vérifier composé de portes élémentaires (ET, OU, PAS, NI, ON) interconnectées, en un nouveau logigramme qui réalise une fonction booléenne prenant la valeur 1 si la combinaison appliquée est un test pour la panne à détecter et la valeur 0 dans le cas contraire. Par opposition au D-Algorithm elle conduit à une procédure très simple à mettre en oeuvre.

La seconde partie présente une méthode de génération des tests pour un circuit cellulaire bidimensionnel permettant de réaliser de façon simple et économique toute fonction booléenne. S'appuyant sur les principes classiques de commande et d'observation, on aboutit à une solution graphique très simple pour détecter les pannes simples ou multiples. On illustre ainsi une méthode adaptée à une réalisation particulière.

2.1. METHODE DE TRANSFORMATION DE LOGIGRAMME

2.1.1. Théorie de la méthode

Considérons le circuit C, figure 2.1. Il a n entrées $X = (x_1, x_2, \dots, x_n)$ et p sorties $Z = (z_1, z_2, \dots, z_p)$. Soit $Z = F(X) = (f_1, f_2, \dots, f_p)$ l'expression générale des p sorties en fonction des variables X . Si m est une connexion quelconque du circuit (voir figure 2.1.), Z peut se mettre sous la forme d'une fonction des $n + 1$ variables (X, m) . Soit $Z = G(X, m) = (g_1, g_2, \dots, g_p)$ l'expression générale de la fonction Z où m est une fonction des variables X dont une expression est $m = m(X)$.

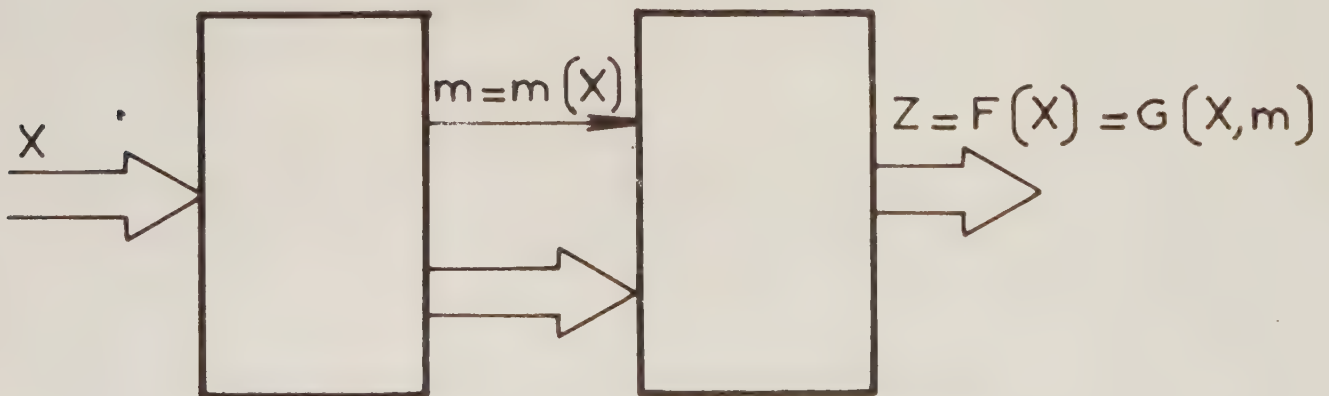


Figure 2.1. Test pour $m/\bar{\alpha}$

Recherchons par exemple un test pour la panne m collée à $\bar{\alpha}$ ($m/\bar{\alpha}$). D'après le théorème 1.1. sur la différence booléenne, ce test est solution de l'équation booléenne suivante :

$$G_m^+ \cdot m^\alpha = 1$$

où G_m^+ est le condensé additif de G_m , différence booléenne de G par rapport à m et m^α égal à m si $\alpha = 1$, à $\bar{m}$ si $\alpha = 0$.

En explicitant le premier membre de cette équation, il vient :

$$(G(X,m) \oplus G(X,\bar{m}))^+ \cdot m^\alpha = 1.$$

Au lieu d'effectuer le calcul sur les expressions booléennes il est possible d'obtenir directement un circuit représentant cette équation.

La figure 2.2.a. schématise le logigramme d'un circuit qui réalise la fonction $Z = G(X,m)$. En reproduisant le même circuit et en remplaçant m par $\bar{m}$ on obtient le logigramme d'un circuit produisant à la fois $G(X,m)$ et $G(X,\bar{m})$. La fonction $m(X)$ et les variables Y sont obtenues par un même logigramme C_1 . Les sous circuits C_2 et C_2' ont la même structure d'interconnexion et ne diffèrent que par l'entrée correspondant à m . (figure 2.2.b.). Nous les distinguerons par un nom différent pour simplifier l'exposé.

Pour compléter la représentation, sous forme de logigramme, de l'équation de départ, il est nécessaire d'effectuer le ou-exclusif, composante à composante des deux vecteurs $G(X,m)$ et $G(X,\bar{m})$ puis le produit de m^α avec le condensé additif. Ce qui nous conduit au logigramme schématisé par la figure 2.2.c.

Le problème de la recherche d'un test pour la panne $m/\bar{\alpha}$ se réduit donc au calcul d'une combinaison des entrées primaires X qui rende la sortie du logigramme figure 2.2.c. égale à 1. Déjà le calcul d'un seul point (ou d'un monôme) d'une fonction réalisée par un logigramme est beaucoup plus rapide sur calculateur que la détermination de chaque terme de l'équation de départ.

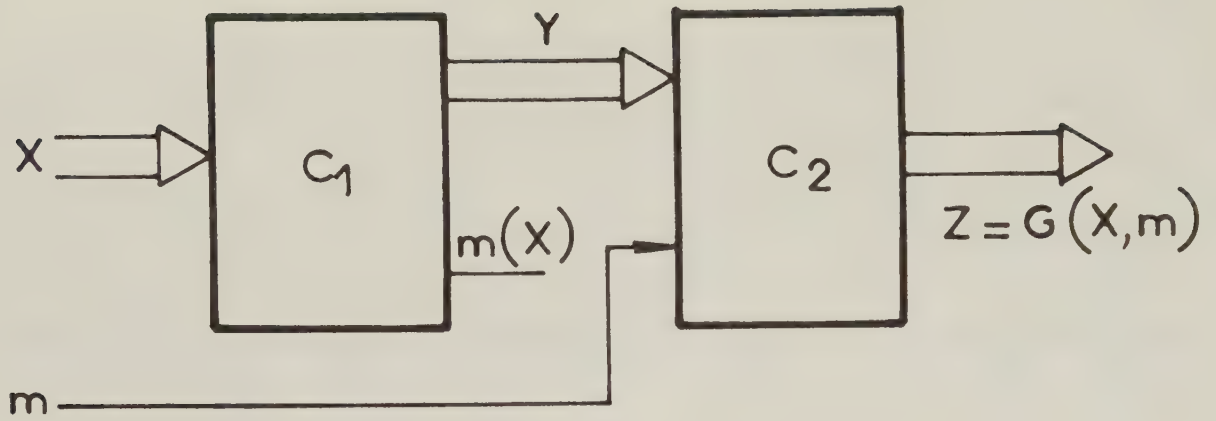


Figure 2.2.a.

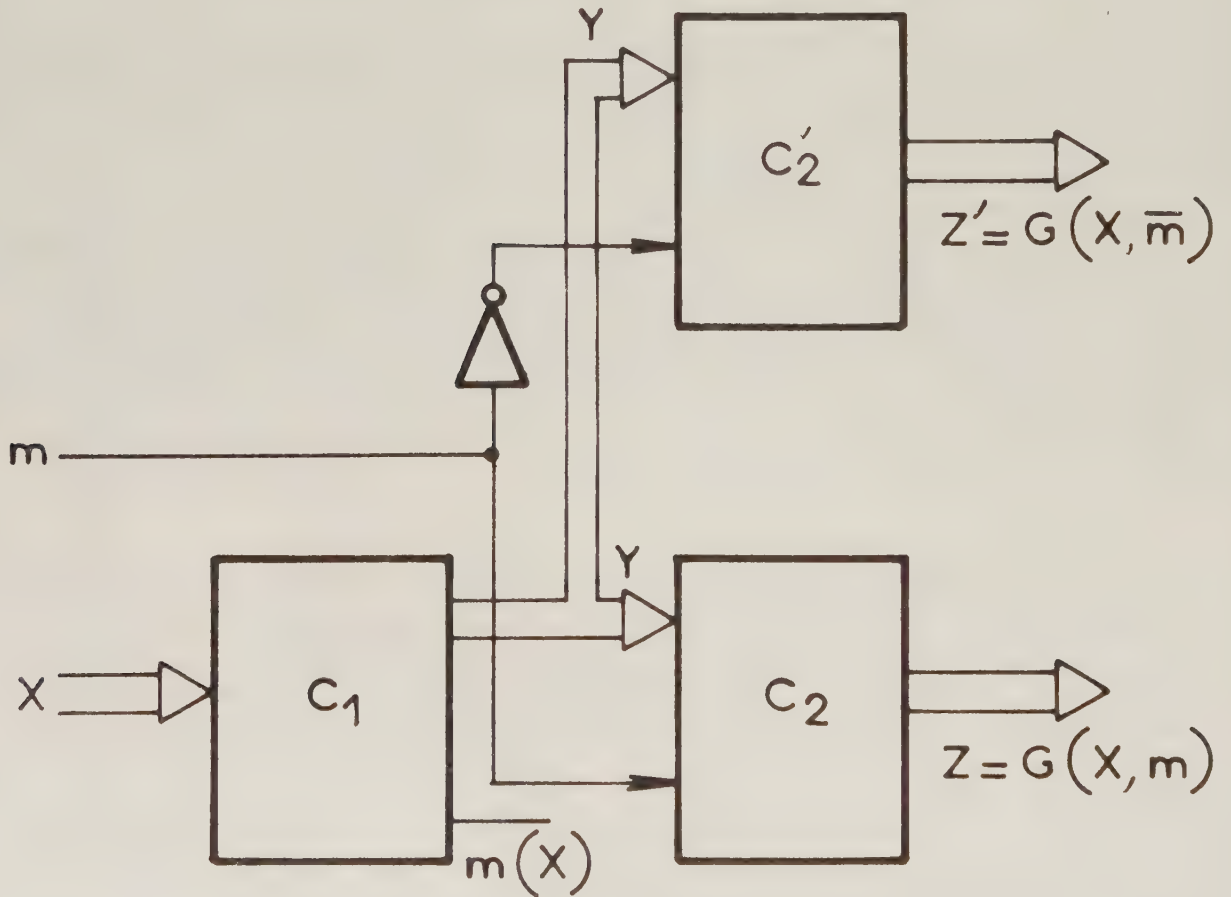


Figure 2.2.b.

Logigramme réalisant $G(X, m)$ et $G(X, \bar{m})$

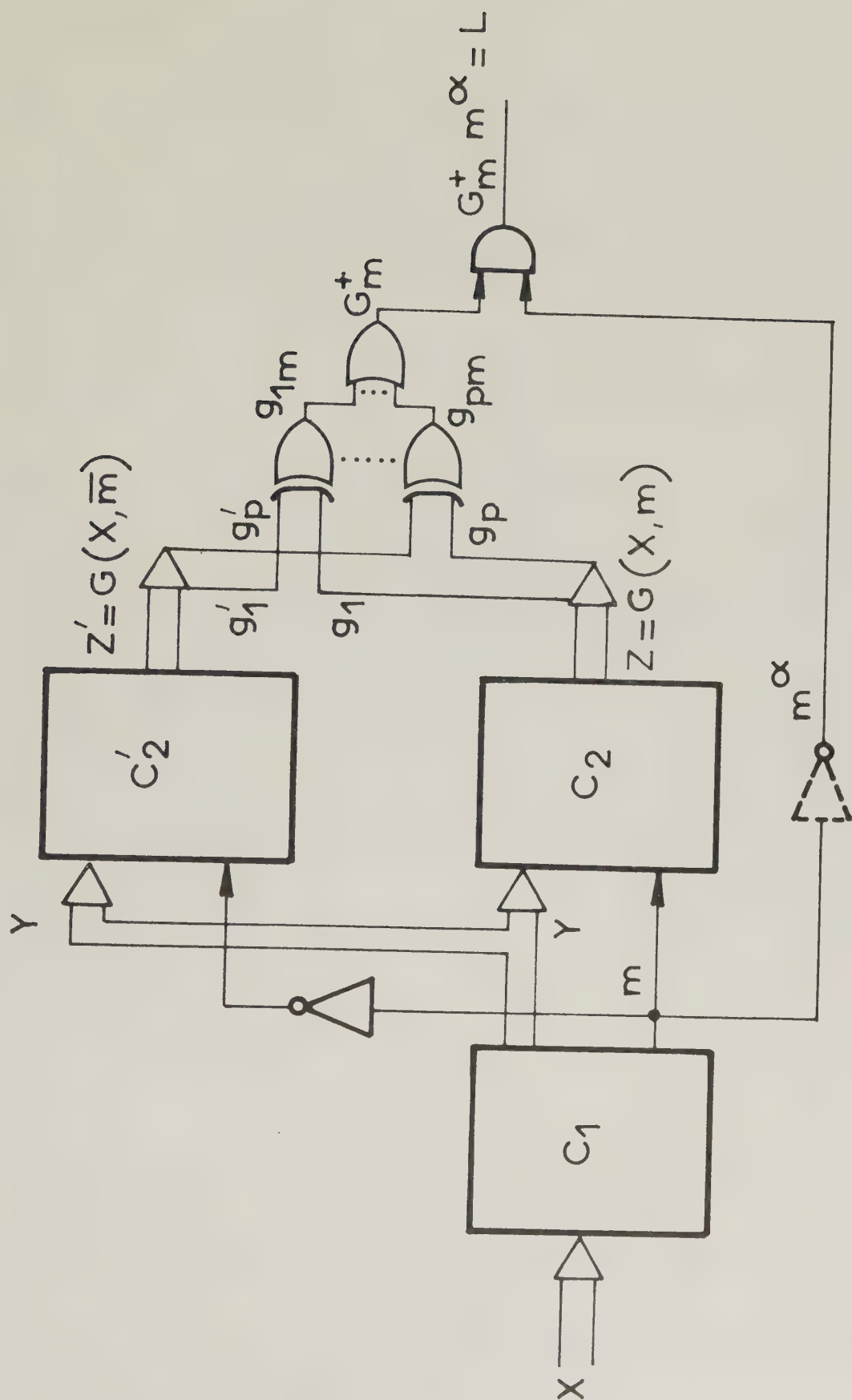


Figure 2.2.c. Logigramme pour la fonction $G_m^+ \cdot m^\alpha$

Une remarque très simple permet de simplifier encore le logigramme, figure 2.2.d.

Dans ce but, analysons la partie C_2 (C'_2) de ce logigramme. Elle se décompose en deux sous-ensembles C_2^1 et C_2^2 ($C_2^{1'}$ et $C_2^{2'}$) définis de la manière suivante :

- C_2^1 est l'ensemble des connexions et des portes successeurs de la connexion m .
- C_2^2 est l'ensemble des connexions et des portes complémentaires de C_2^1 par rapport à C_2 .

Cette décomposition est schématisée sur les figures 2.3.a. et 2.3.b.

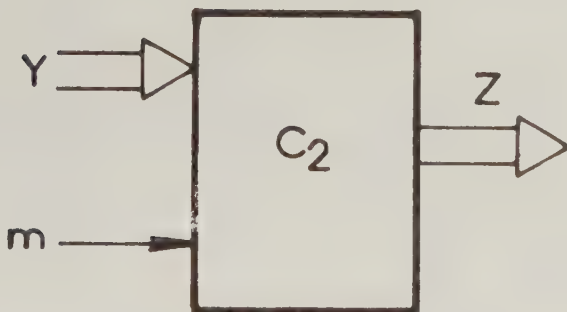


Figure 2.3.a.

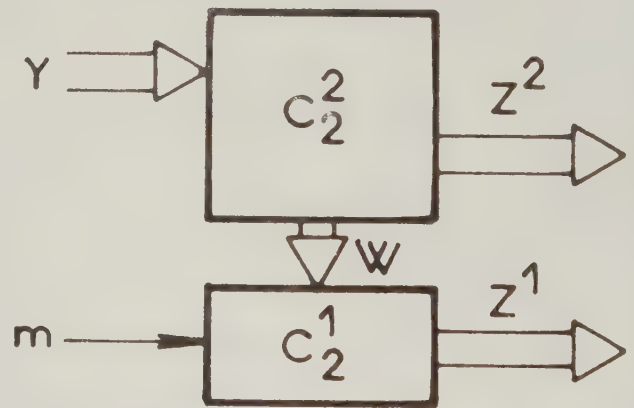


Figure 2.3.b.

Figure 2.3. Ecoulement de la connexion m .

La figure 2.3.b. met en évidence ces deux sous-ensembles qui induisent une partition en Z^1 et Z^2 de l'ensemble des sorties Z . Nous noterons W l'ensemble des connexions de C_2^2 attaquant au moins une porte de C_2^1 . Si l'on fait apparaître cette décomposition sur le logigramme de la figure 2.2.d. on obtient celui de la figure 2.4. Les deux sous logigrammes C_2^2 et $C_2^{2'}$ ont par

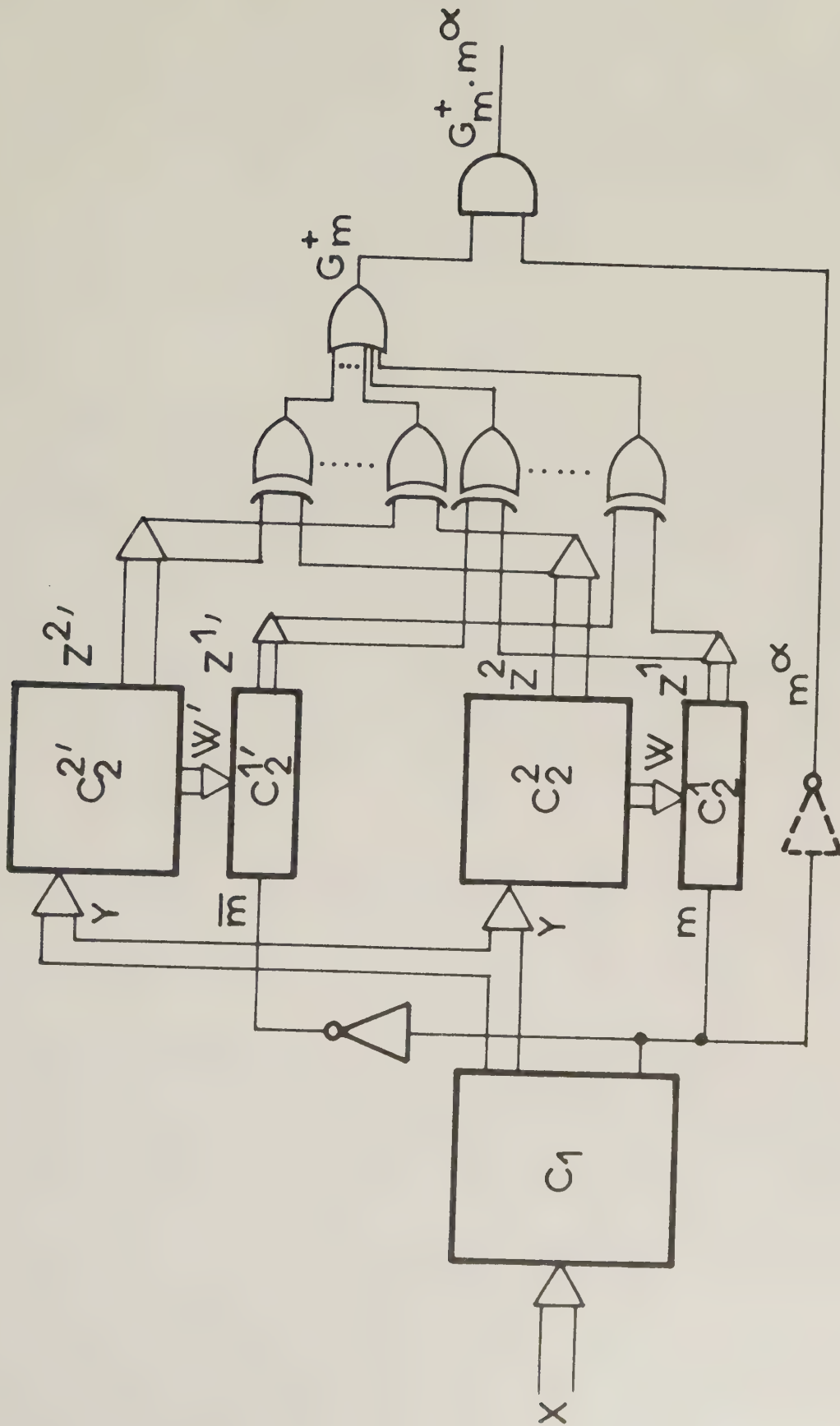


Figure 2.4.

construction la même topologie et les mêmes entrées donc $Z^2 \oplus Z^{2'} = 0$ pour toute combinaison d'entrée, ce qui revient à dire que $G_m^+ = (Z^1 \oplus Z^{1'})^+$. Ainsi l'équation $L = G_m^+ \cdot m^\alpha$ est matérialisée par un logigramme plus simple obtenu en regroupant C_2^2 et $C_2^{2'}$ et ne faisant intervenir que les sorties appartenant à C_2^1 dans le condensé additif G_m^+ . On aboutit alors à un circuit schématisé par la figure 2.5.

On peut encore poursuivre le raisonnement sur le logigramme de la figure 2.5. pour retrouver la plupart des algorithmes de propagation.

Considérons les deux sous circuits C_2^1 et $C_2^{1'}$. Ils ont la même structure les mêmes entrées sauf m qui attaque C_2^1 et $\bar{m}$ $C_2^{1'}$, donc quelle que soit la combinaison d'entrée X_0 appliquée au circuit de la figure 2.5. C_2^1 et $C_2^{1'}$ seront attaqués par des vecteurs adjacents ne différant que par l'entrée m .

Théorème 2.3. Une condition nécessaire et suffisante pour que la combinaison X_0 soit un test pour le collage de $m/\bar{\alpha}$ est qu'il existe au moins une sortie appartenant à Z^1 (soit z^1) et la sortie correspondante dans $Z^{1'}$ (soit $z^{1'}$) qui portent des valeurs booléennes opposées lorsque l'on applique X_0 à l'entrée du circuit figure 2.5.

En effet, dans le cas contraire tous les termes de G_m^+ sont nuls et le circuit, figure 2.5., fournit une sortie égale à zéro. Comme le circuit, figure 2.5., est caractéristique des tests pour la panne $m/\bar{\alpha}$ cela impliquerait que X_0 ne serait pas un test. $\square$

Corollaire 2.2. Une condition nécessaire et suffisante pour que X_0 soit un test est qu'il existe au moins deux chemins homologues mz^1 et $mz^{1'}$ appartenant respectivement à C_2^1 et $C_2^{1'}$ et portant des valeurs booléennes complémentaires pour chaque couple de connexions homologues.

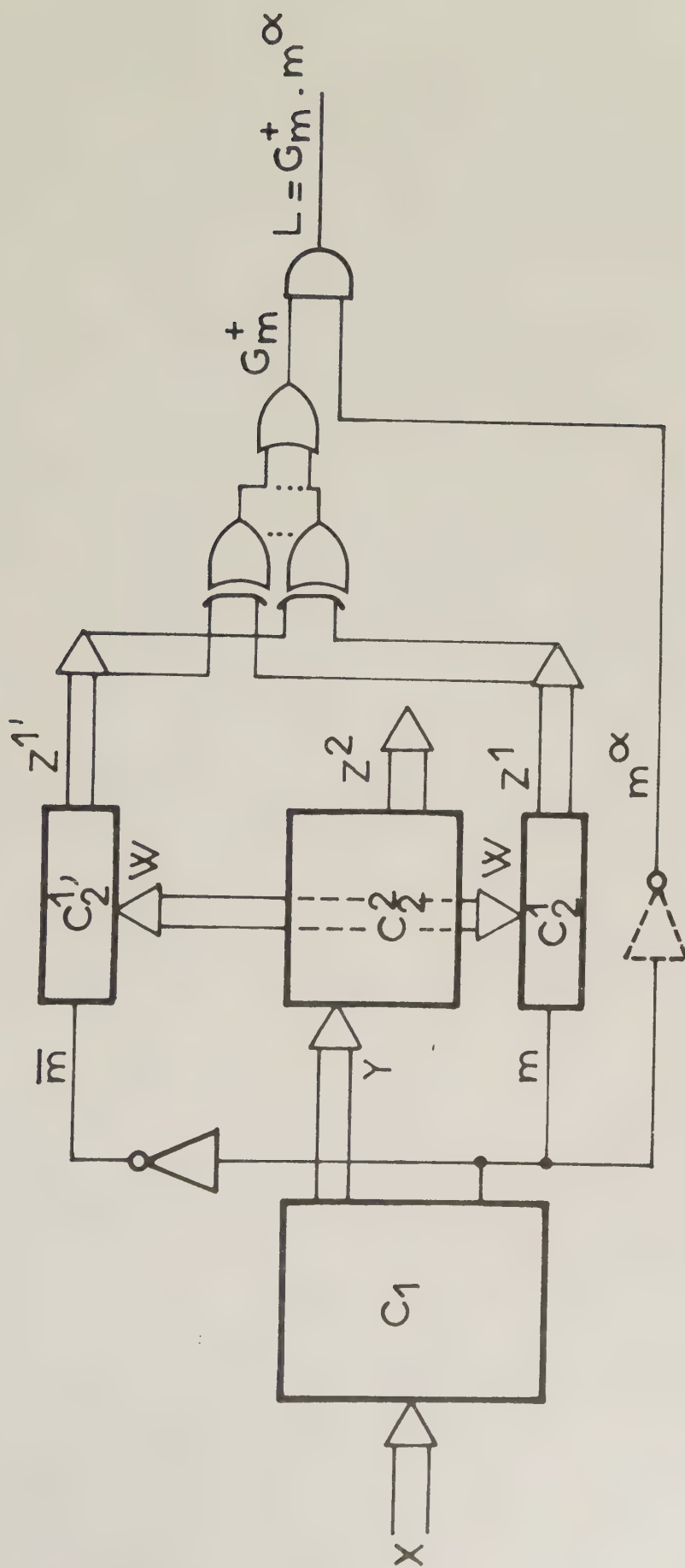


Figure 2.5.

En effet, s'il n'en était pas ainsi, on pourrait extraire de C_2^1 et $C_2^{1'}$ deux sous circuits homologues, ayant en particulier la même structure, les mêmes entrées, auxquels serait appliquée la même combinaison d'entrée et qui fourniraient sur les sorties homologues z^1 et $z^{1'}$ des valeurs booléennes complémentaires. $\square$

C'est pratiquement à l'aide de ce corollaire que sont bâtis tous les algorithmes de recherche des tests par propagation. Leur point commun est qu'il s'agit de procédures d'énumération. En particulier le D-Algorithm correspond à la solution la plus sophistiquée et conduit en un nombre d'essais minimum à une solution. Mais, il est bien connu que pour les procédures d'énumération le nombre minimum d'essais ne conduit pas toujours au temps minimum puisqu'il est alors nécessaire de conserver et de se référer à une grande quantité d'informations.

2.1.2. Solution programmée

Nous avons donc choisi de développer la procédure la plus simple qui correspond à la recherche d'une combinaison X_0 rendant la sortie du circuit, donné par figure 2.5., égale à 1.

Le programme comporte 3 étapes pour la recherche d'un test pour un collage donné du circuit traité, soit m la connexion pour laquelle on recherche un test.

a. Détermination de l'ensemble des connexions et des portes successeurs de m : $C_2^{1'}$

b. Construction du circuit réalisant la fonction $L = G_m^+ \cdot m^\alpha$

c. Calcul d'un point ou d'un monôme de la fonction L .

Nous ne détaillerons pas les algorithmes utilisés. Pour les étapes a et b, il s'agit de solutions classiques de traitement des graphes. L'étape c sera explicitée au paragraphe suivant à propos d'un exemple simple.

NUMBRE D ENTREES 14
NUMBRE DE PERIODES 61
NUMBRE DE SERIES 8

EXEMPLE NUMBRE 1
PANNES DU CIRCUIT 181

NUMBRE DE BRANCHES MULTIPLES 93

TEMPS MOYEN PAR TEST 36 TEMPS TOTAL 1824

VMAX= 530.000 VMIN= 1.000 TCTAL=1824.000 MCY= 36.480 ETV= 106.057

KINF= C KSUP= 2

1	-251	C
2	-221	C
3	-190	C
4	-160	C
5	-130	C
6	-99	C
7	-69	C
8	-39	C
9	-2	41 *****
10	21	3 ***
11	51	1 *
12	81	2 **
13	112	C
14	142	1 *
15	172	C
16	203	C
17	233	C
18	263	C
19	294	C
20	324	C
21	354	C

Figure 2.7. Histogramme des temps de calcul par la méthode de transformation de logigramme.

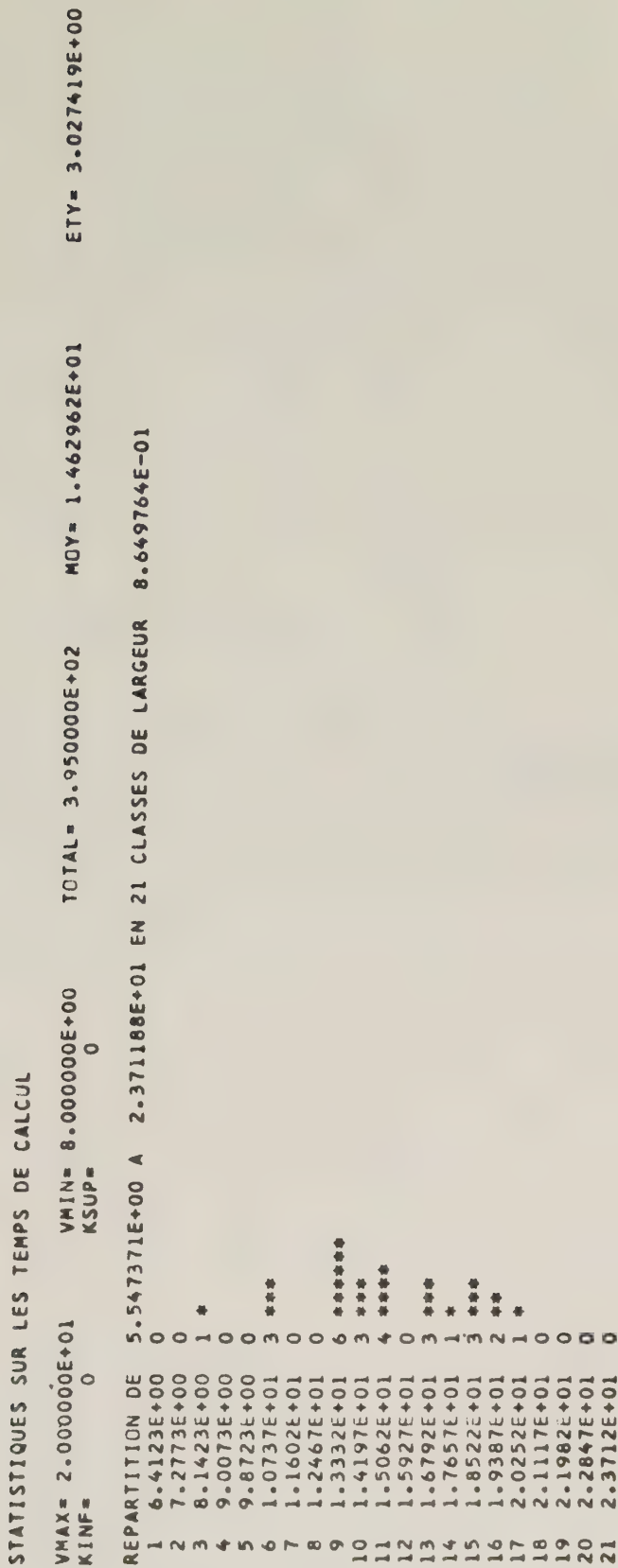


Figure 2.8. Histogramme des temps de calcul à l'aide de Dédale.

circuit 74181

STATISTIQUES SUR LES TEMPS DE CALCUL

VMAX= 9.400000E+01 VMIN= 1.000000E+01 TOTAL= 1.075000E+03 MCV= 3.596665E+01 ETY= 1.877725E+01
 KINF= 0 KSLF= 1

REPARTITION DE -2.036509E+01 A 9.229841E+01 EN 21 CLASSES DE LARGEUR 5.364929E+00

1	-1.5000E+01	0	
2	-9.6352E+00	0	
3	-4.2703E+00	0	
4	1.0946E+00	0	
5	6.4555E+00	0	
6	1.1824E+01	1	*
7	1.7189E+01	5	*****
8	2.2554E+01	3	***
9	2.7919E+01	2	**
10	3.3284E+01	2	**
11	3.8649E+01	5	*****
12	4.4014E+01	6	*****
13	4.9379E+01	2	**
14	5.4744E+01	1	*
15	6.0109E+01	0	
16	6.5474E+01	0	
17	7.0839E+01	1	*
18	7.6204E+01	1	*
19	8.1569E+01	0	
20	8.6934E+01	0	
21	9.2298E+01	0	

Figure 2.9. Histogramme de temps de calcul à l'aide de Dédale.

Nous donnons figures 2.6. et 2.7. les histogrammes des temps de calcul des tests pour les connexions d'un circuit. Seulement 20 % des pannes possibles sont tirées au hasard cela nous permet d'évaluer les temps de calcul à 10 % comme nous avons pu le vérifier sur tous les exemples traités.

Il est intéressant de comparer ces histogrammes avec ceux des figures 2.8. et 2.9. obtenues à l'aide du programme Dédale (TAB 1) qui est une amélioration du D-Algorithm. On remarque que les moyennes obtenues donnent des temps équivalents mais que la courbe obtenue par Dédale est beaucoup plus plate que la courbe obtenue par la méthode de transformation de circuit.

2.1.3. Exemple d'application

Considérons le circuit figure 2.10.a. et recherchons un test pour la connexion m collée à 1. La figure 2.10.b. donne l'ensemble des successeurs de m (circuit C_2^1) et la figure 2.10.c. est le logigramme ; la sortie devient égale à un si et seulement si la combinaison appliquée à ce logigramme est un test pour la panne m collée à 1.

Pour obtenir un point vrai de la fonction réalisée par ce logigramme, on effectue le calcul en remontant depuis la sortie jusqu'aux entrées primaires. On choisit à priori pour chaque porte rencontrée l'une des combinaisons des trois valeurs (0,1,X) à assigner à ses entrées pour satisfaire la valeur portée par la sortie. Les autres combinaisons permettant de satisfaire cette valeur sont conservées en mémoire afin de pouvoir y revenir si l'on aboutit à un moment donné à une incompatibilité.

Le calcul pour le test de la panne $m/1$ est porté par le logigramme, figure 2.10.c. Pour chaque porte, on a représenté les différents assignements d'entrées possibles en choisissant à priori celui représenté le plus à droite. Le test obtenu est alors $(a\ b\ c\ d\ e\ f) = (X\ 1\ 1\ X\ 0\ X)$.

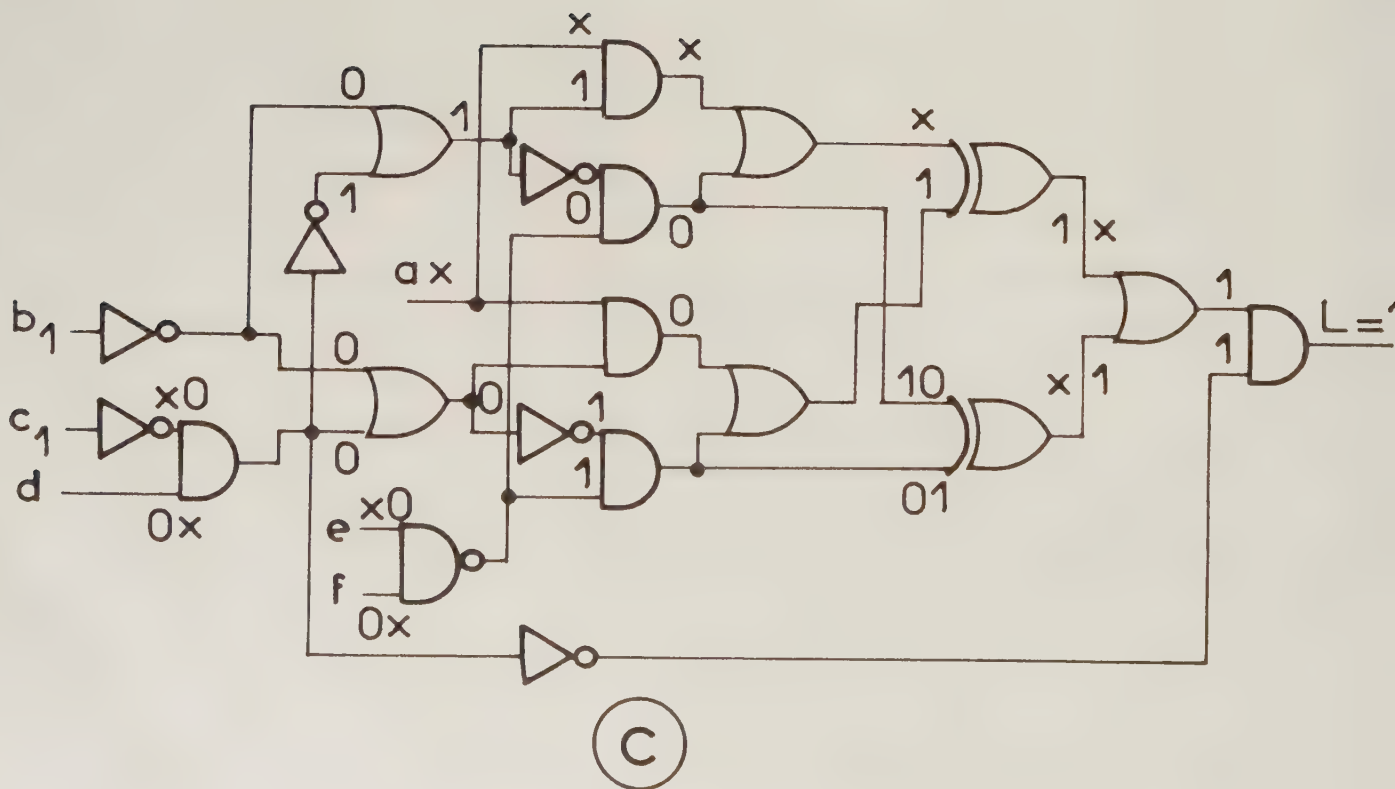
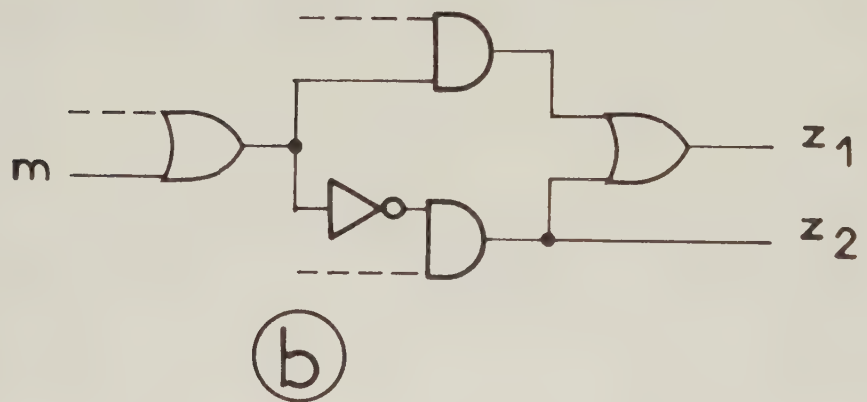
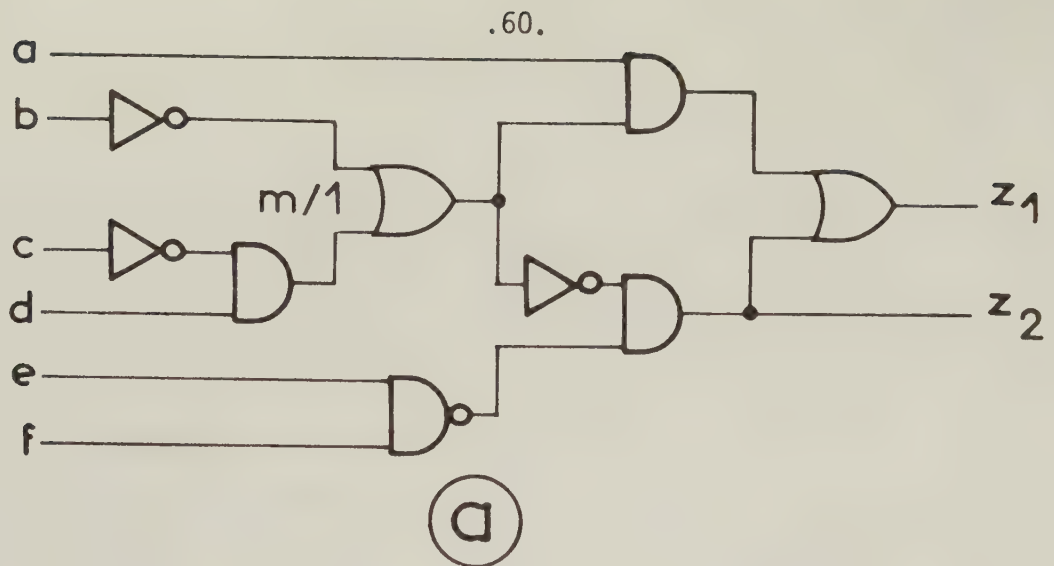


Figure 2.10. Un exemple de recherche de tests par la méthode de transformation de logigramme.

2.2. METHODE GRAPHIQUE SUR UN RESEAU CELLULAIRE

Le développement rapide de la microélectronique a dégagé de nouveaux critères de synthèse tels que régularité, planéarité et facilité de test. De nombreux travaux (MUK 1), (IPP 1) ont montré que les réseaux cellulaires permettent de satisfaire les deux premiers critères. La synthèse par réseaux cellulaires nécessite des choix à priori de structure et de cellule de base. Aussi existe-t-il un grand nombre de solutions proposées. Cette partie n'a donc pas pour objectif de présenter une méthode valable pour tous les cas mais de dégager à travers un exemple particulier le type de procédure que l'on peut mettre en oeuvre en adaptant les notions classiques de commande-observation à une réalisation.

Nous rappellerons brièvement (paragraphe 2.2.1.) les caractéristiques du réseau traité qui est dû à (IPP 1), puis nous présenterons la méthode pour obtenir les tests de détection.

2.2.1. Structure et cellule de base du réseau

Il s'agit d'un réseau composé de cellules identiques interconnectées pour former une grille rectangulaire, figure 2.11. Cette structure permet de décrire le circuit au moyen de trois paramètres qui sont :

sa hauteur h

sa largeur w

le nombre de ses entrées primaires $C = h + w + hw$

En effet, la cellule de base (figure 2.12.) réalise sur deux sorties la fonction $f = \gamma\beta + \bar{\gamma}\alpha$ où α , β et γ sont les trois entrées. Dans (IPP2), on trouve une méthode de synthèse qui permet d'obtenir de façon simple à l'aide de ce réseau toute fonction logique d'un nombre quelconque de variables. Par rapport aux solutions proposées dans (MIN 1), (MIN 2) et (CAN 1), on montre (IPP 2) que le réseau obtenu a des propriétés intéressantes.

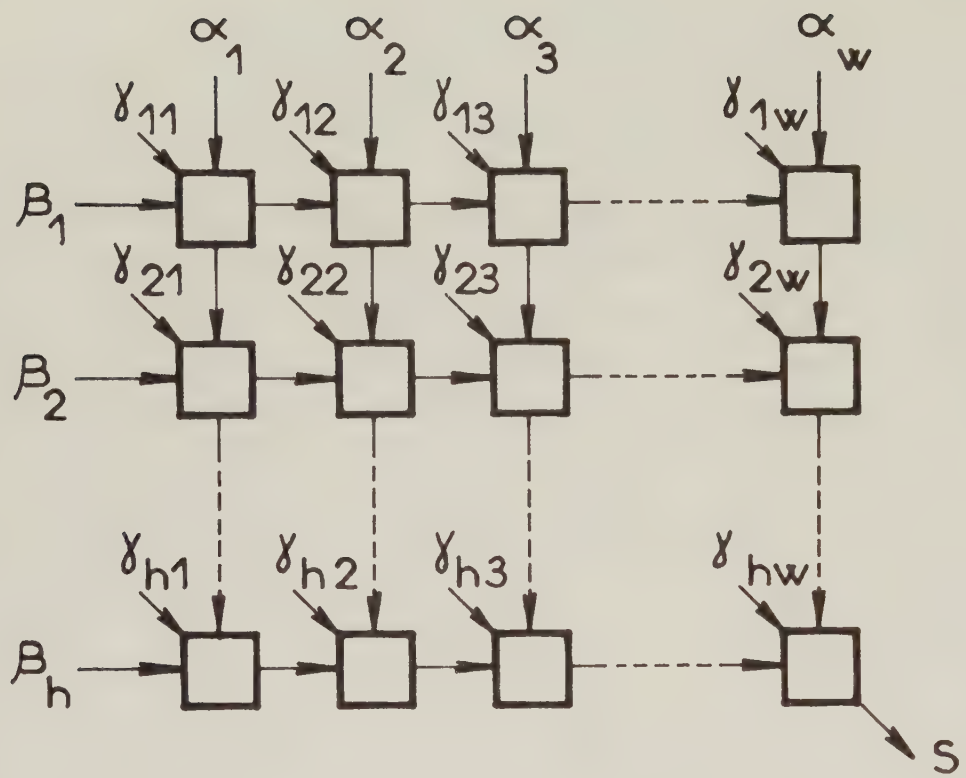


Figure 2.11. Structure du réseau cellulaire

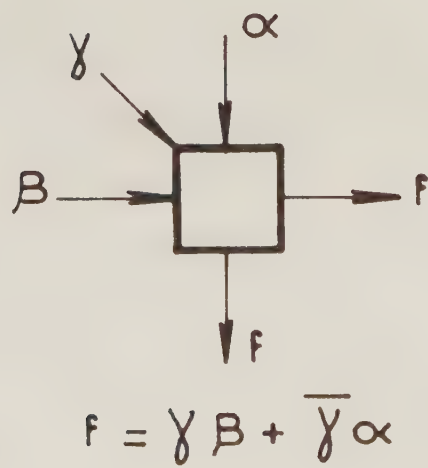


Figure 2.12. La cellule de base

2.2.2. Détection des pannes dans ce type de réseau

a. Propriété de transfert des cellules

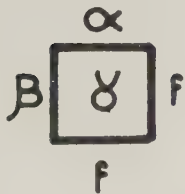
Définitions 2.1. L'une des propriétés intéressantes de cette cellule est qu'elle peut être utilisée comme cellule de transfert de l'information.

.transfert de type T : Si nous assignons la valeur 1 à l'entrée γ , les sorties sont égales à β indépendamment de α . Figure 2.13.b.

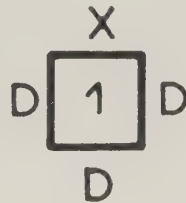
.transfert de type t : Si nous assignons la valeur 0 à l'entrée γ , les sorties sont égales à α indépendamment de β . Figure 2.13.c.

.transfert de type $\wedge$: Si nous assignons les valeurs 0 et 1 respectivement à α et β les sorties sont égales à γ . Figure 2.13.d.

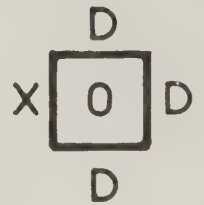
.transfert de type $\overline{\wedge}$: Si nous assignons les valeurs 1 et 0 respectivement à α et β les sorties sont égales à $\overline{\gamma}$. Figure 2.13.e.



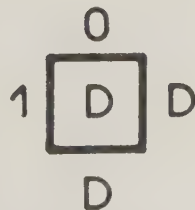
(a) représentation simplifiée de la cellule.



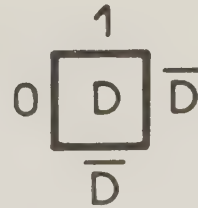
(b) transfert de type T.



(c) transfert de type t.



(d) transfert de type $\wedge$.



(e) transfert de type $\overline{\wedge}$.

Figure 2.13. Transfert d'information à travers la cellule de base.

On peut envisager deux solutions (IPP 2) qui consistent soit à laisser toutes les entrées de type γ indépendantes (paragraphe b) soit à effectuer une métallisation telle que plusieurs entrées de type γ se trouvent reliées à une seule broche accessible (paragraphe c).

Les traitements de ces deux cas étant similaires, nous développerons en détail le premier et nous indiquerons uniquement la marche à suivre pour le second.

b. Toutes les entrées de type γ sont indépendantes

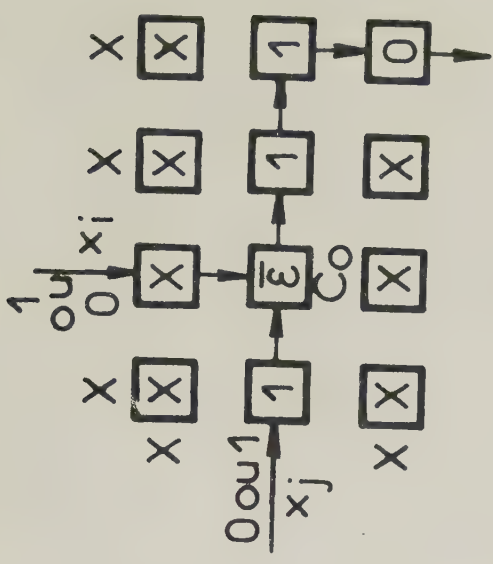
Nous donnerons tout d'abord les conditions que doit vérifier une combinaison d'entrée pour être un test pour le collage d'une connexion quelconque (paragraphe b.1.) puis une méthode graphique simple qui conduit à des ensembles complets de détection pour les collages uniques et multiples.

Génération de tests pour le collage d'une connexion

Afin de détecter un collage simple sur l'entrée $\alpha(\beta, \gamma)$ d'une cellule c quelconque, il est nécessaire de faire apparaître la panne sur les sorties de c (étape appelée commande) c'est-à-dire lui appliquer une combinaison permettant un transfert de type t ($T, \wedge$ ou $\overline{\wedge}$) puis de propager cette information jusqu'à une sortie primaire (étape appelée observation). Comme toutes les entrées γ sont accessibles, on peut choisir un chemin de propagation quelconque.

Théorème 2.3. Soit C_0 une cellule quelconque du réseau. La procédure suivante permet d'obtenir un test pour le collage d'une connexion quelconque attaquant C_0 .

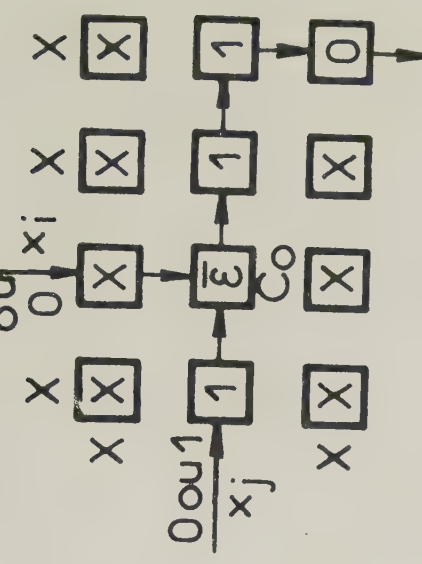
1. Si la panne est un collage à $\mathcal{E}$ de
 - α_{C_0} mettre $\gamma_{C_0} = 0$
 - β_{C_0} mettre $\gamma_{C_0} = 1$
 - γ_{C_0} mettre $\gamma_{C_0} = \overline{\mathcal{E}}$



a. collage à ϵ de α_{C_0}



b. collage à ϵ de β_{C_0}



c. collage à ϵ de γ_{C_0}

Figure 2.14. Tests pour le collage à ϵ d'une connexion quelconque.

2. Choisir un chemin de propagation de f_{c_0} à la sortie primaire et marquer les connexions de ce chemin.

3. Si la panne est sur

- α_{c_0} , choisir un chemin joignant une entrée primaire, soit x_i , à α_{c_0}
- β_{c_0} , choisir un chemin joignant une entrée primaire, soit x_i , à β_{c_0}
- γ_{c_0} , choisir deux chemins joignant respectivement α_{c_0} et β_{c_0} à deux entrées primaires distinctes x_i et x_j . Marquer le ou les chemins choisis.

4. Considérer chaque cellule c différente de c_0 du réseau marqué.

- si c a une entrée verticale marquée, mettre $\gamma_c = 0$
- si c a une entrée horizontale marquée, mettre $\gamma_c = 1$
- si c n'a aucune entrée marquée, mettre $\gamma_c = X$

5. Si la faute est un collage à $\mathcal{E}$ de :

- α_{c_0} mettre x_i à la valeur $\overline{\mathcal{E}}$
- β_{c_0} mettre x_i à la valeur $\overline{\mathcal{E}}$
- γ_{c_0} mettre $(x_i, x_j) = (0,1)$ ou $(1,0)$.

6. Mettre toutes les entrées non assignées à X .

Cette construction est représentée sur les figures 2.14.a., b et c respectivement pour des collages à $\mathcal{E}$ de α_{c_0} , β_{c_0} , γ_{c_0} .

Démonstration :

Pour démontrer que les combinaisons d'entrée obtenues sont bien les tests recherchés, il suffit de montrer d'une part que la connexion à vérifier porte bien la valeur booléenne $\overline{\mathcal{E}}$ et d'autre part que la sortie vaut soit $\mathcal{E}$ soit $\overline{\mathcal{E}}$.

En effet si la panne est sur α_{c_0} ou β_{c_0} chacune des cellules c appartenant au chemin marqué joignant x_i à α_{c_0} ou β_{c_0} est telle que son entrée γ_c porte une valeur booléenne qui rend ses sorties égales à son entrée marquée et indépendante de l'entrée non marquée. Ainsi, la connexion à vérifier porte bien la même valeur que l'entrée primaire x_i . Le même raisonnement est valable en ce qui concerne le chemin joignant f_{c_0} à la sortie primaire, ce qui montre bien dans le cas de collages sur α_{c_0} et β_{c_0} la validité du théorème.

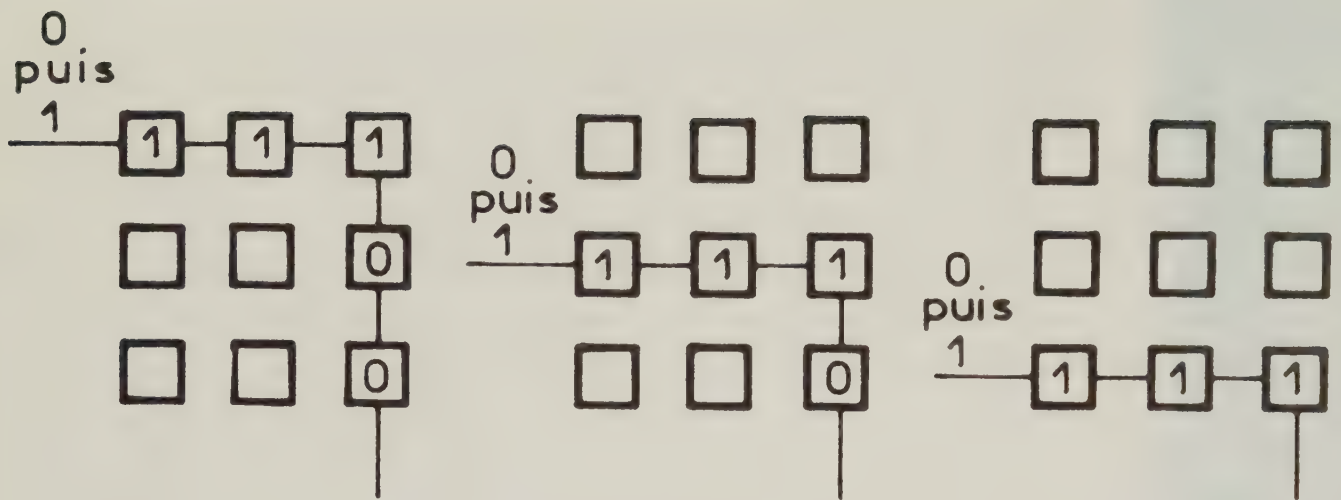
Si la panne est sur γ_{c_0} , cette connexion porte, par construction la valeur $\bar{E}$. Les valeurs affectées à $(x_i, x_j) = (0,1)$ ou $(1,0)$ permettent un transfert de type $\wedge$ ou $\bar{\wedge}$ donc f_{c_0} porte la valeur E ou $\bar{E}$. L'affectation des entrées γ_c du chemin marqué joignant f_{c_0} à la sortie primaire implique que cette dernière porte la même valeur booléenne que f_{c_0} , ce qui complète la démonstration. $\square$

b. Génération d'un ensemble de tests de détection

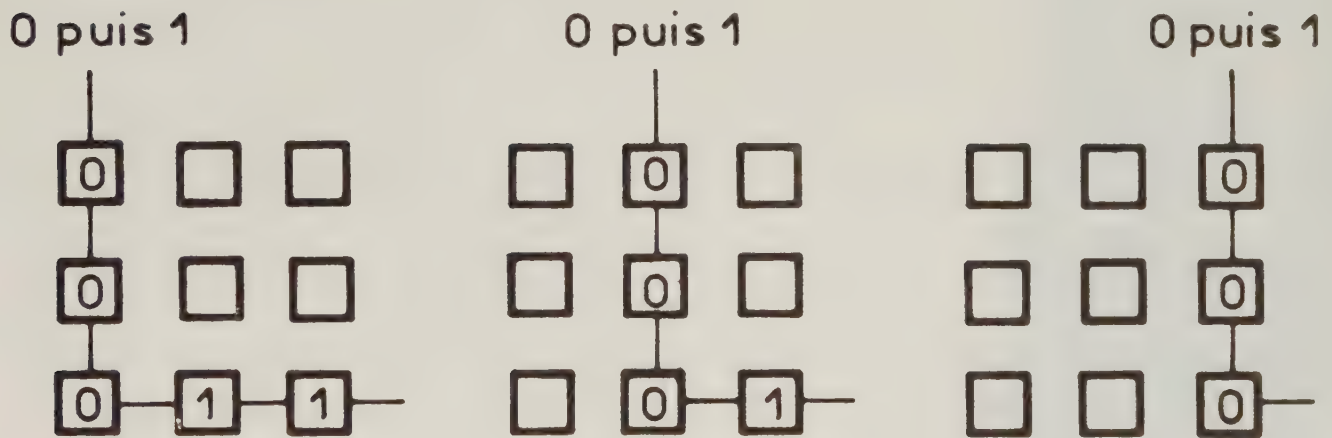
Ensemble de détection pour les collages simples

Lorsque l'on génère par cette méthode un test pour une entrée primaire de type α ou β il est évident que toutes les connexions α ou β appartenant au chemin marqué sont vérifiées pour le même type de collage. En effet, comme nous l'avons vu au cours de la démonstration du théorème 2.3., elles portent toutes la même valeur $\bar{E}$ par construction.

Pour vérifier tous les collages des connexions α et β , il suffit donc d'appliquer la procédure précédente aux entrées primaires en choisissant les chemins marqués pour qu'ils constituent une couverture de toutes les connexions α et β du réseau. Ce choix est toujours possible, il suffit pour cela d'utiliser, par exemple, des chemins marqués de type $\neg$ (antigamma) et L , définis ci-après.



a. chemins de type 7



b. chemins de type L

Figure 2.15. Couverture de l'ensemble des connexions de type α et β par des chemins de type L et 7.

Définition 2.2. Les figures 2.15.a. et b définissent respectivement les chemins γ et L réalisant la couverture de toutes les connexions α et β .

Théorème 2.4. L'ensemble des tests obtenus en prenant pour chemins marqués ceux de type L et γ et en appliquant sur l'entrée primaire correspondante 0 puis 1 est un ensemble de détection pour toutes les pannes simples du réseau, si l'assignement des cellules n'appartenant pas au chemin marqué ne varie pas entre les deux tests associés à ce chemin.

Nous avons vu précédemment que cet ensemble de tests, puisque les chemins marqués sont une couverture de l'ensemble des connexions α et β détectait toute panne sur les connexions α et β . Il suffit donc de prouver que cet ensemble détecte aussi les pannes uniques sur les entrées de type γ .

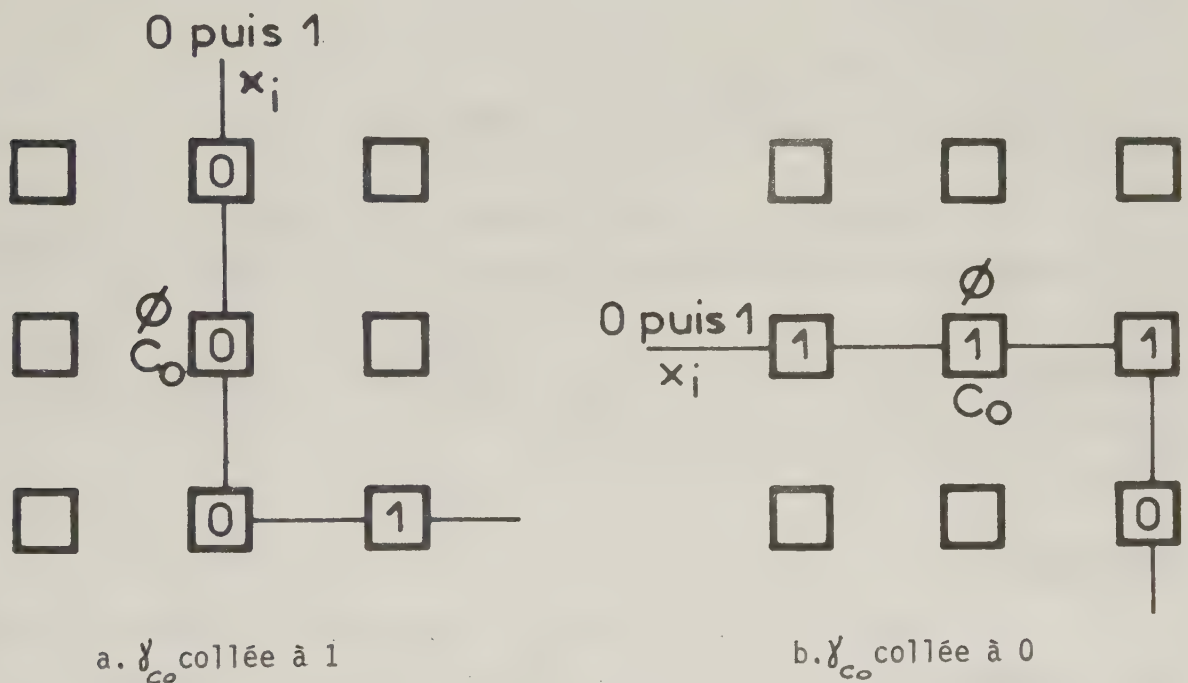


Figure 2.16. Les tests pour γ_{C_0}

Considérons une cellule C_0 quelconque et un chemin de type $L(\gamma)$ passant par C_0 et tel que $\gamma_{C_0} = 0(1)$ figures 2.16.a.(et b). Supposons que la valeur portée par la connexion $\beta_{C_0}(\alpha_{C_0})$ n'appartenant pas au chemin marqué ne change pas entre les deux tests associés à ce chemin, soit $\emptyset$ cette

valeur, l'entrée x_i prend successivement les valeurs 1 puis 0. Donc, pour l'un des deux tests, on trouve sur les entrées $\beta_{C_0}, \alpha_{C_0}$ ($\alpha_{C_0}, \beta_{C_0}$) la combinaison $(\emptyset, \emptyset)$.

Pour cette combinaison particulière un transfert de type $\wedge$ (ou $\bar{\wedge}$) et le chemin de propagation jusqu'à la sortie primaire existent. C'est donc un test pour γ_{C_0} collée à 1(0). Ce qui démontre la propriété. $\square$

Ensemble de détection pour les collages multiples

Définition 2.3. On appelle chemin de type $\neg^{10}$ (L^{10}) un chemin $\neg(L)$ tel que pour toute connexion d'entrée w n'appartenant pas au chemin marqué:

- a. si w est au-dessous et/ou à gauche du chemin alors $w = 1$
- b. si w est au-dessus et/ou à droite du chemin alors $w = 0$.

La figure 2.17. donne l'ensemble des chemins de type L^{10} et $\neg^{10}$ constituant une couverture de toutes les connexions α et β du réseau.

Théorème 2.5. L'ensemble des tests correspondants à une couverture des connexions α et β par des chemins de type $\neg^{10}$ et L^{10} est un ensemble de détection pour les pannes multiples du réseau.

Démonstration :

Considérons une panne de collage multiple. Elle est constituée d'un ensemble de collages simples de connexions de type α , β et γ . Supposons tout d'abord que la panne multiple considérée affecte au moins une connexion γ et montrons que dans ce cas elle est détectée.

Lemme 2.1. Toute panne multiple contenant un collage à zéro sur une connexion γ d'une cellule appartenant à la ligne p d'un réseau de hauteur p est détectée par les tests correspondant à $\neg_p^{10}$ (figure 2.18.).

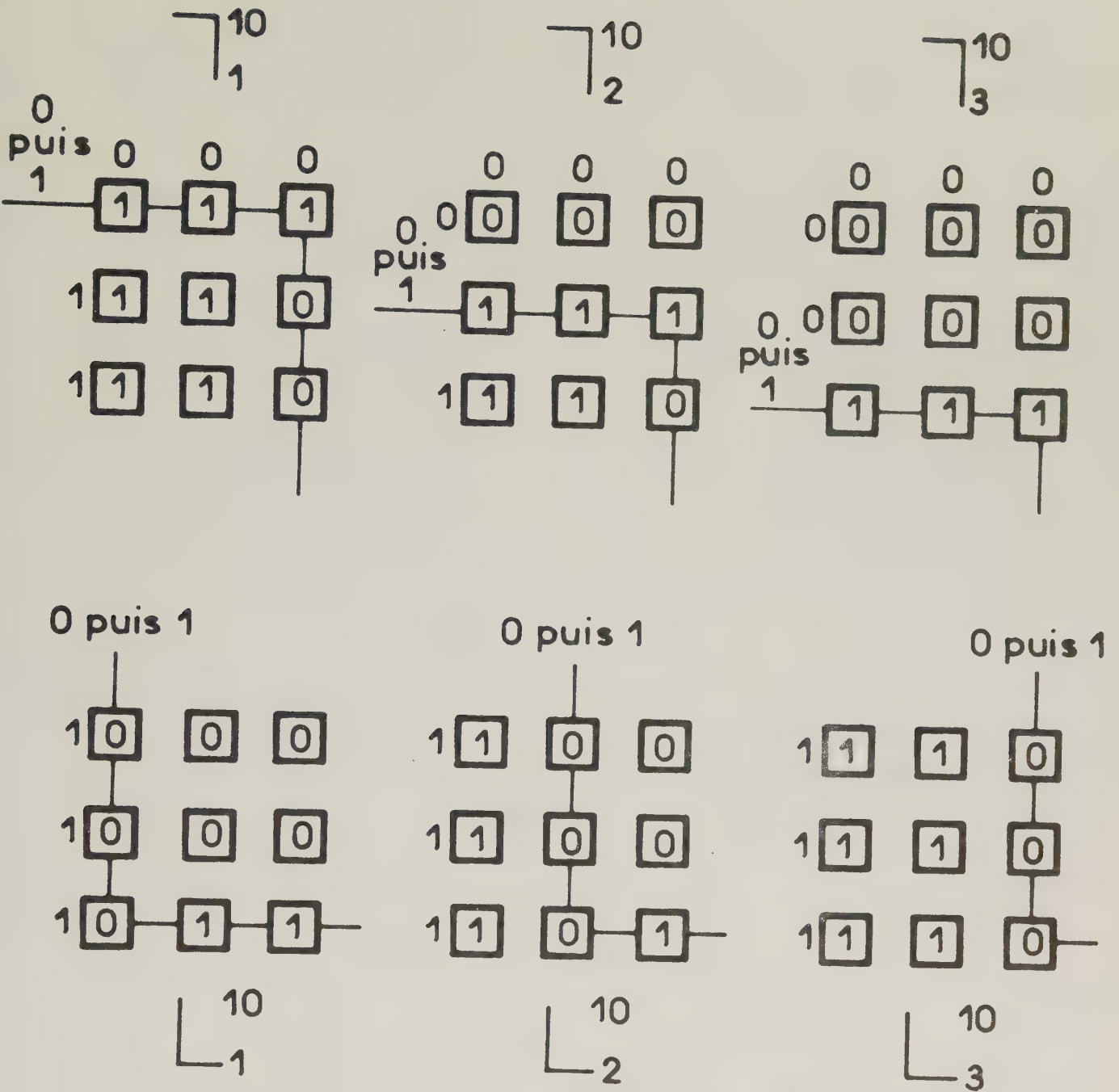


Figure 2.17. Couverture des connexions α et β par des chemins γ^10 et L^10 .

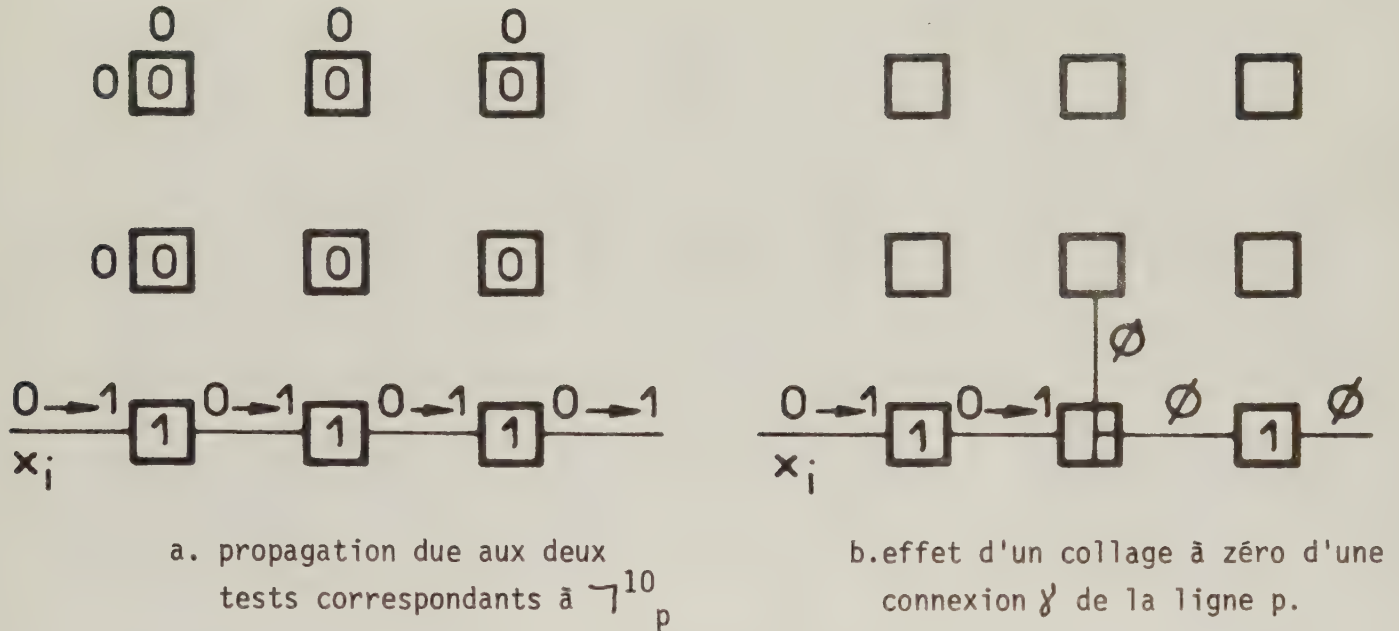
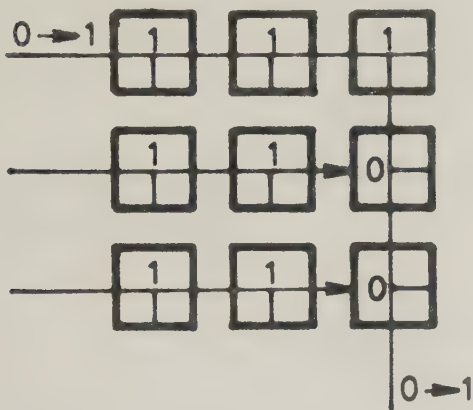


Figure 2.18. Détection d'un collage multiple contenant le collage à zéro d'une connexion de la ligne p.

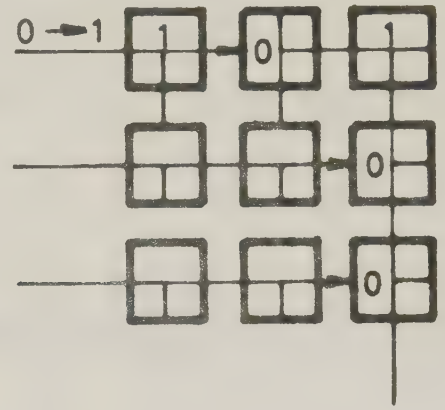
En effet, toutes les cellules de la ligne p sont utilisées lors de ces tests, comme cellules de transfert de type T. Un collage à zéro de l'une des connexions γ transforme la cellule correspondante en transfert de type t. Lorsque le réseau n'a aucune panne, la variation de l'entrée x_i se propage le long de la dernière ligne jusqu'à la sortie primaire. La panne sur l'entrée γ a pour effet de bloquer la propagation. Comme il n'existe aucun autre chemin possible, la sortie reste à une valeur fixe entre les deux tests, ce qui permet de détecter la panne (figure 2.18.a. et b.) $\square$

Supposons maintenant que toutes les lignes depuis la ligne $i+1$ jusqu'à la ligne p sont telles que leurs connexions γ ne sont pas collées à zéro et montrons alors que si une panne multiple contient un collage à zéro d'une connexion γ de la ligne i les tests associés à γ_i^{10} permettent de le détecter.

En effet, la variation de l'entrée x_i correspondante est bloquée au niveau de la première cellule ayant sa connexion γ collée à 0 et ne peut être propagée par aucune autre cellule vers la sortie primaire puisque toutes les cellules de la ligne suivante ($i+1$) sont de type T et ne peuvent pas être collées à 0 par hypothèse. La figure 2.19.a. montre la propagation à travers le circuit correct. La figure 2.19.b. montre le blocage de la variation au niveau d'une cellule de type T. Ainsi aucune des connexions γ ne peut être collée à zéro sans être détectée. $\square$



a. propagation de la variation



b. blocage par une panne.

Figure 2.19.

Lemme 2.2. Toute panne multiple contenant le collage à un sur une connexion γ d'une cellule appartenant à la q ième colonne d'un réseau de largeur q est détectée par les tests correspondants à $L^{10}q$.

La démonstration de ce lemme, puis un raisonnement par récurrence similaire aux précédents conduit donc à montrer qu'aucune connexion γ ne peut être collée sans être détectée. $\square$

Il reste à considérer le cas où la panne multiple affecte seulement des connexions α et β .

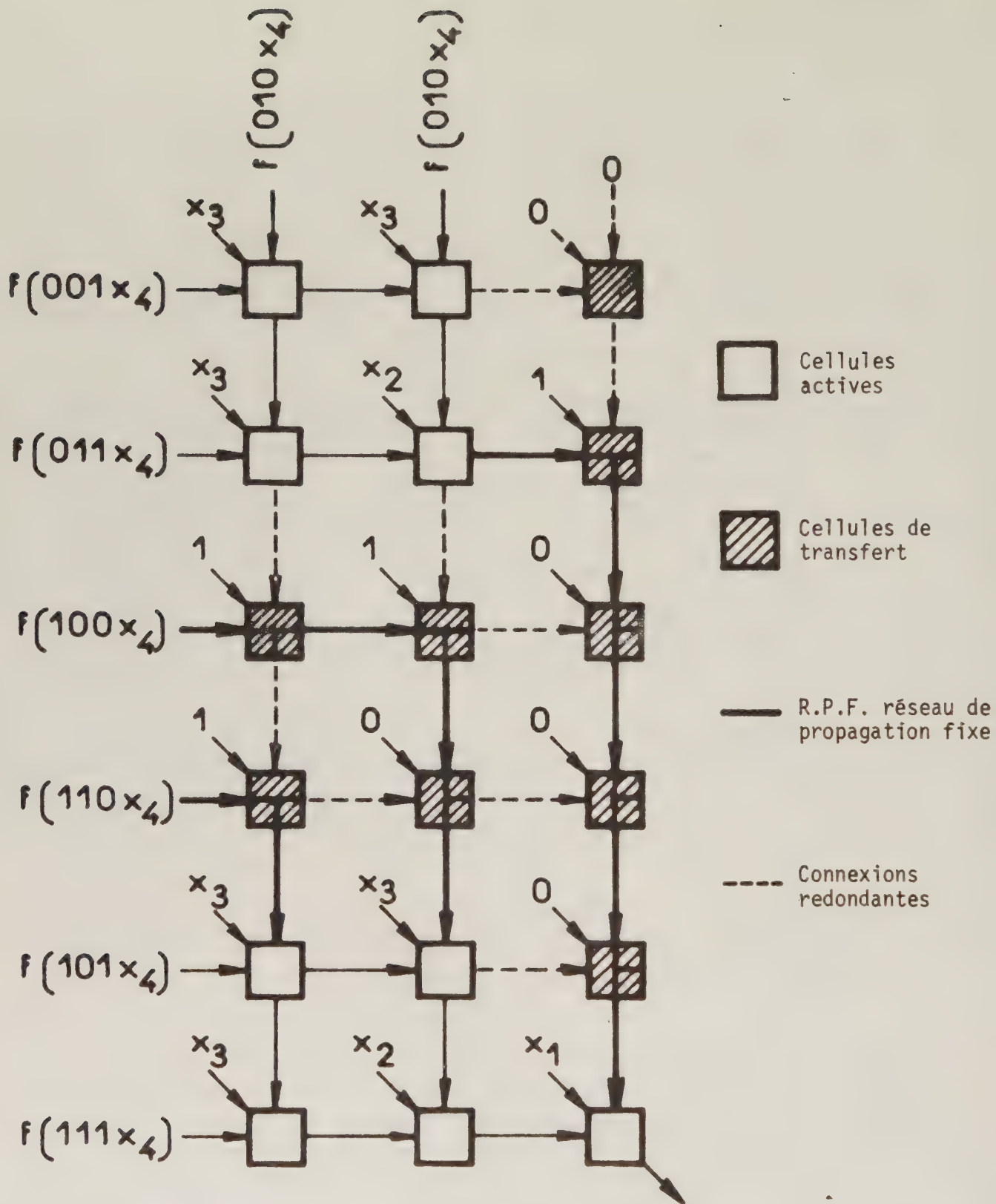


Figure 2.20. Réseau universel pour les fonctions de 4 variables.

Propriété 2.3. Toute panne sur les connexions α et β n'a aucun effet si elle n'affecte pas le chemin marqué.

En effet, puisque les connexions γ sont supposées sans panne la valeur portée par une connexion du chemin marqué est indépendante des valeurs portées par les connexions n'appartenant pas à ce chemin.

Propriété 2.4. Un ensemble de pannes sur les connexions α et β a le même effet que la panne simple qui est la plus proche de la sortie primaire sur le chemin marqué.

Ces deux propriétés terminent donc la démonstration du théorème 2.5. $\square$

c. Certaines entrées γ peuvent être connectées

Certaines entrées primaires de type γ peuvent être reliées à une même entrée primaire x_i ou à une même valeur logique 0 ou 1.

La figure 2.20. représente une telle réalisation, ce réseau permet de générer toute fonction de quatre variables. On remarque qu'il existe alors un ensemble de cellules de transfert (hachurées sur la figure 2.20.) qui définissent un réseau de propagation fixe (R.P.F.). L'existence du R.P.F. induit un ensemble de connexions α et β dont la valeur logique est sans importance pour le comportement du réseau (en pointillé sur la figure 2.20.).

Certaines entrées γ étant dépendantes, il n'est pas possible de choisir une couverture quelconque des connexions du réseau. Il est nécessaire dans ce cas de construire une couverture de tout le réseau compatible avec le R.P.F.

Il est ainsi possible de montrer que cet ensemble de tests permet de détecter toute panne unique du réseau.

2.3. CONCLUSIONS

Nous avons présenté dans ce chapitre, deux méthodes utilisant comme modèle de fonctionnement le logigramme du circuit. Leur avantage essentiel est d'éviter le calcul toujours long de fonctions booléennes et d'aboutir ainsi, de manière relativement rapide, à un test pour une panne donnée.

Ces méthodes sont donc spécifiques de la réalisation et devront être adaptées si l'on désire traiter d'autres types de réalisation ou d'autres réseaux cellulaires.

La première partie s'applique à un logigramme de portes élémentaires ET, OU, PAS, NI, ON. Les temps de calcul obtenus sont intéressants, par exemple 35/100 de seconde pour obtenir un test pour une panne donnée d'un circuit d'environ 60 portes élémentaires sur un calculateur IBM 360/168.

La seconde partie, utilisant les notions classiques de commande et d'observation, conduit à une solution graphique très simple adaptée à la structure particulière d'un réseau cellulaire bidimensionnel. Elle peut s'appliquer manuellement à des réseaux de très grande dimension.

Pour traiter les systèmes séquentiels, la solution généralement retenue (KUB 1), (GIA 1) est une extension des méthodes combinatoires. Elle consiste à construire un modèle itératif, encore appelé modèle spacio-temporel, dont chaque élément représente un instant pour le circuit.

On le considère alors comme un ensemble combinatoire auquel on impose des contraintes. En effet, il faut que la séquence obtenue soit applicable à partir de n'importe quel état interne. Ainsi que le souligne (GIA 1) il ne s'agit que d'une méthode heuristique, c'est-à-dire que les séquences obtenues ne sont pas avec certitude des séquences de test, mais plutôt des "tests potentiels" pour le circuit réel. Il est ensuite néces-

saire de les valider à l'aide d'un simulateur tenant compte de paramètres plus fins, en particulier les retards associés à chaque élément.

Bien que la recherche d'un test soit une procédure plus rapide sur un logigramme, les algorithmes sont en grande partie à base d'énumération. La taille des circuits à traiter étant toujours plus importante, on aboutit à une limite des possibilités de ces solutions. Il n'est pas réaliste de penser résoudre le problème des systèmes logiques complexes à l'aide de telles solutions.

Le problème de la taille des circuits paraît soluble en utilisant une description de la structure de réalisation sous forme de macrocomposants pour lesquels on aurait déjà calculé les tests. Cette solution qui, dans une certaine mesure, améliorera certainement les temps de calcul ne nous semble pas lever tous les inconvénients des méthodes purement structurelles. En effet, les séquences de vérification obtenues par la méthode du modèle itératif ne tiennent aucun compte des contraintes imposées par le concepteur quant au domaine de fonctionnement du système. On peut donc obtenir des séquences quelconques. Comme le modèle de fonctionnement n'est en général valable qu'à l'intérieur du domaine de fonctionnement normal (en particulier parce qu'on ne tient aucun compte des retards), on assiste à un pourcentage de rejet des séquences potentielles générées qui est une fonction fortement croissante de la taille du circuit (ROT 1).

L'introduction comme paramètre de la génération des tests des retards associés aux éléments conduit à des algorithmes trop complexes pour traiter des circuits de grande dimension. Cela nous a amené à rechercher une nouvelle description d'un système logique permettant de prendre en compte, au niveau de la génération des tests, le fonctionnement normal du système. Le chapitre 3 est consacré à la définition et l'utilisation de ce nouveau modèle.

3. DESCRIPTION MISSION - RÉALISATION

UTILISATION POUR LE TEST

Lorsque la taille des systèmes à traiter augmente s'appuyer uniquement sur un modèle de structure comme un logigramme, rend le problème de la détection impossible à résoudre.

Sans nier la nécessité du modèle de structure, nous avons donc pensé utiliser en plus d'autres informations qui sont détenues par le concepteur et qui sont contenues dans une description de la mission du système.

Nous étudions tout d'abord les défauts des modèles classiques utilisés par la détection des pannes. Nous en déduirons les propriétés que doit posséder un modèle de comportement pour faciliter la génération des tests. (paragraphe 3.1.).

Il sera ensuite proposé un nouveau modèle (paragraphe 3.2., 3.3. et 3.4.) et nous montrerons une utilisation de ce modèle pour la génération d'un ensemble de détection (paragraphe 3.5. et 3A).

3.1. LES DEFAUTS DES MODELES CLASSIQUES : PROPRIETES D'UN NOUVEAU MODELE

3.1.1. Défauts des modèles classiques

Dépendants du type de réalisation

Les modèles classiques ne sont pas appropriés pour générer les tests. En effet, d'une manière générale, les méthodes efficaces qui en découlent sont dépendantes du type de réalisation employé.

Cela est particulièrement clair pour les méthodes sur les logigrammes présentées au chapitre 2. Elles traitent de circuits modélisés par une structure d'interconnexion de portes élémentaires ou de cellules de base. Les méthodes algébriques peuvent être appliquées à des réalisations différentes mais sont d'une part très limitées quant à la taille des circuits traités et nécessitent, d'autre part, de connaître les fonctions associées

aux connexions à vérifier. Les fonctions n'étant pas toujours connues, il est nécessaire de concevoir des procédures effectuant ce calcul pour les différentes réalisations à traiter. Notre conviction est qu'au niveau des circuits de base ou primitives d'une réalisation qui sont en général des unités technologiques, il sera toujours nécessaire de concevoir des méthodes et des programmes spécifiques du type de réalisation considéré et du modèle de panne traité.

Il existe d'autres méthodes, utilisant comme modèle le graphe ou la table de fluence pour représenter le fonctionnement d'un système séquentiel qui sont par conséquent sans lien direct avec la réalisation. Elles sont de plus l'avantage de traiter une très grande part des pannes possibles. Malheureusement, leur applicabilité est limitée à une petite classe de systèmes séquentiels qui correspondent à des machines comportant un petit nombre d'états et ayant des propriétés d'observabilité particulières (séquences de distinction et d'entrée-sortie).

Un système complexe est très souvent réalisé à l'aide de sous-ensembles appartenant à des classes différentes, c'est-à-dire dont le fonctionnement est représenté par des modèles différents. S'il est donc possible d'appliquer à chacune des parties une procédure différente, il reste souvent très difficile, à l'aide des modèles classiques d'assembler les résultats obtenus. Certains ont proposé de rechercher pour chaque sous-ensemble un logigramme équivalent et ainsi d'unifier le traitement. Mais les difficultés pour obtenir ces logigrammes et transposer les pannes d'un modèle à l'autre montrent qu'il est assez peu réaliste de penser représenter un système tout entier par un logigramme.

Complexité et temps de calcul

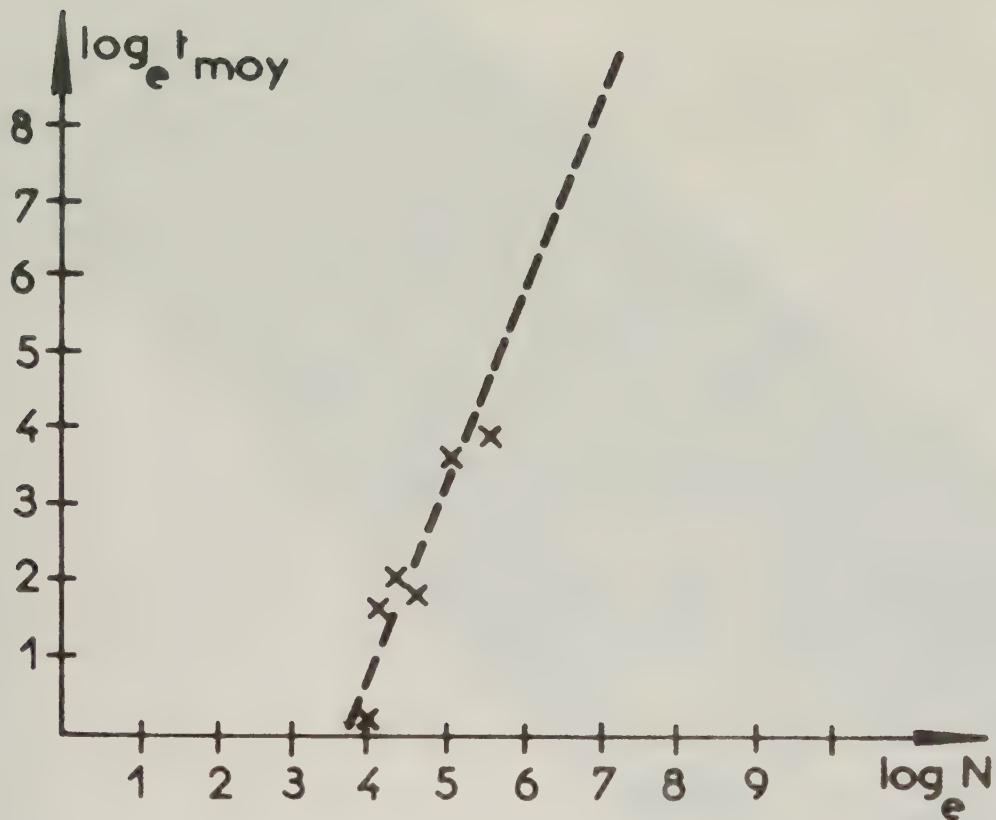
La recherche d'un test est un problème de nature fortement combinatoire. Comme nous l'avons déjà souligné dans les deux premiers chapitres, les méthodes algébriques et les méthodes sur un logigramme sont applicables raisonnablement jusqu'à une certaine taille de circuit. En effet

Le calcul d'un test pour une panne donnée est un problème fortement combinatoire puisque les méthodes de génération sont en grande partie des énumérations implicites : ainsi, on aboutit à des temps de calcul croissant rapidement avec la taille du circuit traité. Tenant compte du fait qu'il n'existe, à notre connaissance, aucun résultat sur les temps de calcul de génération de tests pour des systèmes logiques complexes, nous nous sommes appuyés sur différentes observations et expériences pour prévoir l'ordre de grandeur des temps nécessaires pour obtenir un test sur un modèle en logigramme :

La génération des tests, même pour un circuit combinatoire, contient comme étape obligée la consistance encore connue sous le nom de "satisfiabilité". Ce problème est classé (COO 1), (GAR 1) parmi les problèmes N-P complets, c'est-à-dire une classe de problème que l'on sait, à l'heure actuelle, résoudre uniquement par énumération. Le temps de résolution d'un problème N-P complet est, dans l'état actuel des connaissances, fortement suspecté de ne pas être borné par kN^α où N est le nombre de bits nécessaires pour coder le problème et α une constante mais par une fonction exponentielle de la forme $k \exp(N)$. Nous poserons donc que le temps de calcul pour générer un test est une fonction exponentielle de la taille du circuit à traiter représentée par son nombre de connexions.

En utilisant ce fait, on peut alors extrapoler les résultats que nous avons obtenus à l'aide des programmes de génération des tests DEDALE (TAB 1) et TRANS (RIC 4) pour obtenir une courbe donnant une évaluation des temps de calcul en fonction de la taille du circuit. On obtient une droite dans le plan temps-moyen pour générer un test (en centièmes de seconde), nombre de connexions du circuit.

On peut enfin analyser les résultats publiés sur la simulation des pannes dans les systèmes logiques complexes. La simulation d'un défaut sur une structure de réalisation est toujours beaucoup plus rapide que la recherche d'une séquence de test pour le même défaut puisque dans le premier cas il s'agit simplement d'une analyse de fonctionnement alors que le second



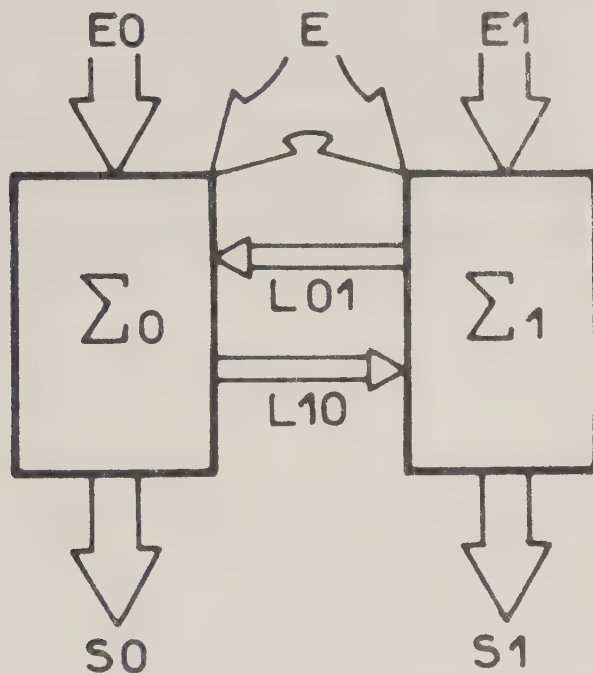
problème est un processus de synthèse. On doit donc considérer que les temps obtenus en simulation sont très largement des minorants des temps espérés pour la génération. Voici à titre d'exemple une table des temps de calcul obtenus sur IBM 360/67 lors de la simulation de listes de défauts sur différents logigrammes supposés sans retard de propagation (ARM 1). Il faut remarquer que ces temps sont considérés comme de bons résultats sur la simulation de défauts.

NOMBRE DE PORTES	NOMBRE DE PANNES	NOMBRE DE VECTEURS	TEMPS DE SIMULATION EN HEURES
6.250	13.310	48	0,33
1.940	2.930	600	2,97
4.820	7.820	755	4,28
3.030	6.000	1.500	5,08
6.660	9.730	1.360	19,04
13.080	38.350	510	21,03
10.960	19.080	1.310	42,02
19.480	45.450	750	60,09

Ce tableau ne permet pas d'obtenir des résultats quantitatifs ramenés en temps moyen par panne pour un circuit donné. Il montre toutefois clairement que la taille des circuits que l'on peut espérer traiter avec un modèle en logigramme est limitée. Ainsi, on peut estimer raisonnable de traiter par les méthodes algébriques des circuits composés d'une centaine de portes élémentaires et par les méthodes sur le logigramme des circuits allant jusqu'à mille portes élémentaires.

Description d'un sous-ensemble

Il arrive souvent que l'on désire générer les tests de détection pour un sous-ensemble et là encore les descriptions s'appuyant uniquement sur la structure de réalisation présentent plusieurs inconvénients. Considérons le système Σ représenté ci-dessous :



Le système Σ est composé de deux sous-ensembles Σ_0 et Σ_1 , soit par exemple, à décrire le sous système Σ_0 . On peut évidemment extraire Σ_0 de son contexte et le décrire indépendamment de Σ_1 , c'est en particulier ce que l'on fait lorsque l'on modélise un sous système par sa structure de réalisation. Cela conduit lors de la génération des tests à des

séquences ne tenant pas compte des contraintes impliquées par les liaisons entre Σ_0 et Σ_1 . Ces séquences sont donc d'une part plus difficiles à appliquer pratiquement et d'autre part elles conduisent Σ_0 à travers des séquences non prévues lors de la conception. Cela est un point essentiel si l'on se souvient que, bien souvent, le logigramme est un schéma équivalent dont le domaine de validité se limite au fonctionnement normal. Bien sûr, il est toujours possible de décrire l'ensemble du système Σ à l'aide d'une description de type logigramme, mais cela conduit à une complexité inutile. La cause de ces difficultés est la faible complexité des éléments de base ou "primitives" utilisées dans la description de structures de réalisation pour la génération des tests.

Positionnement dans un état de test

En plus de la complexité du problème purement combinatoire de la génération d'un test, l'une des difficultés essentielles de l'obtention d'une séquence pour une panne donnée est de conduire le système dans un état (ou un groupe d'états) à partir duquel ce test peut-être appliqué. Cela provient de la nature même des descriptions classiques qui sont des instantanés qui spécifient la transition d'un état présent vers l'état futur sans expliciter en aucune façon la notion de temps ou de séquence.

Difficulté de compréhension

Enfin, les descriptions classiques possèdent l'inconvénient d'être difficilement compréhensibles. Par exemple, si l'on se donne le logigramme d'un compteur sans spécifier à priori la fonction de ce système, on sera amené à effectuer une recherche de tests et de séquence de positionnement ce qui est manifestement un gaspillage puisque la seule façon de tester un compteur et bien sûr de le faire compter.

3.1.2. Propriétés pour un nouveau modèle de description

Afin de pallier ces difficultés, il est donc nécessaire d'avoir une description possédant les caractéristiques suivantes :

.Comporter une partie totalement indépendante de la technologie pour permettre de traiter de façon unifiée un système logique complexe.

.Posséder comme primitives des transformations permettant de traduire de façon simple les commandes et les opérations usuelles dans les systèmes logiques puisque, d'une manière générale, la complexité de la description d'un système diminue lorsque le langage utilisé admet pour primitives les opérations et les commandes usuelles.

.Permettre de décrire une même transformation à différents niveaux de précision ou de détail, afin de pouvoir "focaliser" sur le sous-ensemble à traiter et traduire l'environnement de façon simple.

.Représenter de façon explicite la notion de temps.

.Etre facilement, ou du moins plus facilement, compréhensible qu'une structure de réalisation.

.Conserver un lien avec la structure de réalisation et cela pour deux raisons :

- d'une part, pour aboutir à des méthodes de tests plus simples. En effet, les méthodes issues de la théorie des automates, par exemple, qui ne tiennent pas compte de la structure de réalisation conduisent à des solutions plus lourdes aussi bien au niveau de la génération des tests que des séquences obtenues puisqu'elles doivent prendre en compte toutes les structures de réalisation possibles ;

- d'autre part, pour pouvoir être utilisables au niveau de la conception d'un système et ainsi ne pas nécessiter un effort supplémentaire.

L'ensemble de ces contraintes ne peut être résolu qu'à l'aide d'un langage de description évolué. Il est d'ailleurs surprenant que les chercheurs préoccupés par le problème du test n'aient pas senti cette nécessité alors que depuis 1950 (BAR 1) les concepteurs, conscients des limites inhérentes aux représentations par fonctions booléennes et par logigrammes, ont porté leurs efforts sur l'élaboration de langages de description évolués. Il semble que cela traduise le souci de rester aussi proche que possible de la réalité technologique en particulier en ce qui concerne la modélisation des pannes. Pourtant, il est bien clair que les logigrammes ne représentent la réalité que lorsque les circuits sont faiblement intégrés et que la notion de panne d'une connexion n'a plus aucune existence physique. Dans le cas où un circuit est à haut niveau d'intégration, il arrive souvent que l'on ne possède même plus le logigramme et s'il existe, il s'agit seulement d'un schéma purement fonctionnel.

Nous présenterons dans une première partie (paragraphe 3.2.) les différents éléments permettant de définir la description de la mission d'un système :

Il s'agit de décrire les transformations subies par des sous-ensembles d'un système logique en réponse à une séquence d'entrée quelconque. Pour cela, nous montrerons qu'une transformation élémentaire peut s'exprimer sous forme symbolique comme l'assignation d'une fonction définie sur un ensemble d'identificateurs I à l'un des éléments de I ce qui s'exprime par $i \leftarrow f(I)$.

La notion de temps se traduit au niveau conceptuel par une relation d'ordre partiel entre les diverses modifications élémentaires que l'on peut associer au système logique. Afin de faciliter la compréhension humaine et le traitement automatique, on effectue alors un ordonnancement de ces modifications. Suivant la séquence appliquée une même modification élémentaire peut être suivie de modifications élémentaires différentes. Cela suggère l'utilisation d'un organigramme comme support graphique.

Cette description de la mission est intimement liée à une description de la réalisation. Ainsi, il est possible de faire correspondre à chaque modification un circuit combinatoire extrait de la réalisation dont la finalité est la matérialisation de cette modification.

Dans une deuxième partie, nous donnerons une définition formelle de la description Mission-Réalisation (paragraphe 3.3.) et nous illustrons par un exemple très simple (paragraphe 3.4.).

Dans la dernière partie, nous montrons l'utilisation que l'on peut faire de cette description pour obtenir un ensemble de tests :

Le modèle de panne traité est la panne unique d'un des éléments de la description de la Mission. Cela conduit à trois conditions nécessaires de commande - mise en évidence et observation. On peut en dégager un ensemble de prétest qui sur les exemples traités s'est révélé particulièrement efficace.

Pour les pannes non détectées par l'ensemble de prétest le choix d'une séquence est facilité par la description de la mission. Le calcul d'un test peut ensuite s'effectuer automatiquement en utilisant les circuits combinatoires extraits de la réalisation et correspondant aux différentes modifications du chemin choisi.

3.2. DESCRIPTION MISSION-REALISATION D'UN SYSTEME LOGIQUE (D.M.R.)

3.2.1. Définition de la partie Mission d'une D.M.R.

La partie mission d'une D.M.R. ou description fonctionnelle est une procédure qui à toute séquence d'entrée appartenant au fonctionnement normal fait correspondre un ordonnancement des séquences de modifications que subissent les blocs fonctionnels mémoires et sorties de cette réalisation, cet ordonnancement ayant la propriété de ne changer aucun des résultats fonctionnels.

Nous nous attacherons tout d'abord à bien faire sentir les différentes notions introduites dans cette définition.

3.2.2. Identificateurs de la description : blocs fonctionnels

D'une manière générale, il s'agit de décrire le comportement d'un système logique en réponse à des séquences d'entrée. Pour cela, il est nécessaire d'utiliser au moins trois ensembles de variables représentant les entrées, les éléments mémoire ou variables d'état et les sorties de ce système.

La première étape de cette description est donc la définition de chacun de ces ensembles ainsi que le domaine des valeurs prises par leurs éléments. Pour les descriptions usuelles -fonctions booléennes ou logigrammes- il s'agit d'ensembles scalaires.

Ici, par contre les éléments peuvent être des scalaires, des vecteurs ou des matrices dont on précise les dimensions. Bien qu'il soit possible d'envisager une description dans laquelle les identificateurs sont définis sur différentes bases de numération, nous avons préféré, afin de permettre une automatisation plus aisée de la génération des tests, limiter le domaine de définition à l'ensemble $\{0,1\}^n$ où n est la dimension de l'identificateur.

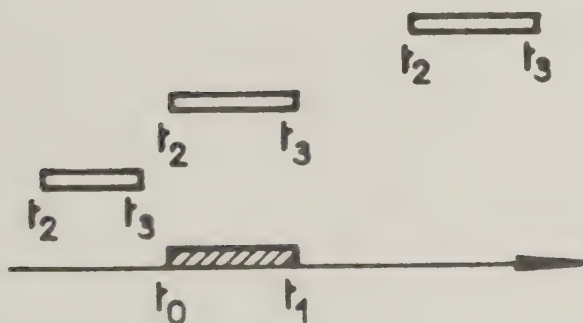
La partition des ensembles scalaires entrées, sorties et mémoires en blocs est en fait l'un des points clef de la description. Ces blocs ne correspondent pas toujours à des unités "circuits" (quoique bien souvent une bonne conception tende à réaliser cette correspondance entre fonction et technologie). Il est donc particulièrement difficile de donner une définition précise d'un bloc fonctionnel. Il s'agit d'éléments mémoires, entrées, sorties groupés sous forme scalaire vectorielle ou matricielle dans le simple but de clarifier la description du comportement en la rendant plus globale. En fait, la détermination des blocs fonctionnels nécessite une

collaboration étroite avec le concepteur du système car ils correspondent en général aux variables qu'il a choisis (registre, compteur...).

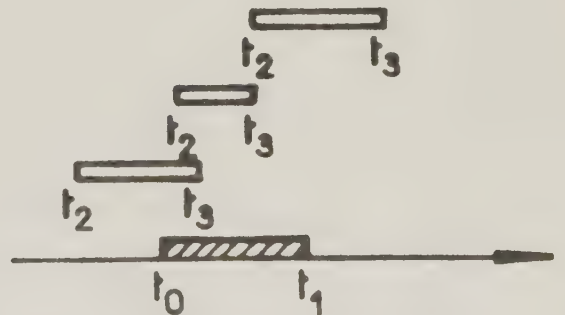
On peut caractériser toutefois un bloc fonctionnel par une propriété temporelle. Un bloc fonctionnel est un sous-ensemble d'entrées, d'éléments mémoires ou de sorties tel que les modifications de deux éléments de ce bloc peuvent toujours être considérées comme se produisant de manière totalement ordonnées.

Explicitons sur un exemple simple, cette notion d'ordre sur les modifications. Comme nous le verrons, une modification s'exprime généralement par l'affectation d'une expression algébrique d'identificateurs à un autre identificateur. Par exemple $C \leftarrow A+B$; $D \leftarrow E \wedge F$. Les opérations sont en pratique matérialisées par un circuit. La durée d'exécution de ces modifications dépend du circuit et éventuellement des valeurs des opérandes. Si t_0 , t_1 et t_2 , t_3 sont les dates de début et de fin d'exécution respectivement des première et seconde modifications, nous dirons qu'elles sont ordonnées, si pour toutes les valeurs de A, B, C et D on peut considérer soit que $t_2 < t_3 < t_0 < t_1$ soit que $t_0 < t_1 < t_2 < t_3$ soit que $t_0 = t_2$ et $t_1 = t_3$.

On schématise ci-dessous les cas de modifications ordonnées et les cas de modifications asynchrones.



modifications ordonnées



modifications asynchrones

Chaque identificateur est donc désigné par un groupe de caractères dont les indices sont précisés par $1 : N$ ou $1 : N$.

Manifestement, la définition de blocs fonctionnels est fortement dépendante du type d'opérateurs dont on dispose pour décrire les modifications que subissent les valeurs portées par les identificateurs. En effet, si l'on considère un ensemble de mémoires utilisées en registre à décalage il n'est intéressant de les déclarer en tant que vecteur que si l'on dispose de primitives permettant de traduire globalement cette transformation, ou encore si cette transformation peut s'exprimer simplement à l'aide de primitives du langage.

3.2.3. Opérateurs de la description

Les identificateurs contiennent donc l'information sous forme de bits. Toute activité du système se traduit par une transformation des valeurs contenues dans ces identificateurs. Les opérateurs sont les transformateurs qui permettent d'exprimer toute modification.

En fait, il est clair qu'un ensemble complet d'opérateurs booléens, par exemple ET, OU, PAS, est suffisant pour traduire toute modification au niveau scalaire.

Toutefois, comme nous l'avons souligné précédemment, la facilité pour décrire le comportement d'un système réside en grande partie dans la possibilité de traduire toute modification -en particulier les modifications vectorielles- à l'aide d'une primitive du langage.

Nous avons donc choisi comme langage support A.P.L. pour sa richesse en opérateurs et la facilité qu'il procure pour traiter les éléments vectoriels et matriciels et nous avons donc admis aussi comme limites les règles de langage (IVE 1).

Forme monadique fB			f	Forme dyadique AfB										
Définition ou Exemple	Nom		Nom	Définition ou Exemple										
$\rightarrow B \leftrightarrow 0+B$	Plus		+ Addition	$2+3.2 \leftrightarrow 5.2$										
$-B \leftrightarrow 0-B$	Moins		- Soustr.	$2-3.2 \leftrightarrow -1.2$										
$\times B \leftrightarrow (B>0) -(B<0)$	Signe		x Multipl.	$2 \times 3.2 \leftrightarrow 6.4$										
$\div B \leftrightarrow 1 \div B$	Inverse		÷ Division	$2 \div 3.2 \leftrightarrow 0.625$										
<table><tr><td>B</td><td>⌈B</td><td>⌊B</td></tr><tr><td>3.14</td><td>4</td><td>3</td></tr><tr><td>-3.14</td><td>-3</td><td>-4</td></tr></table>	B	⌈B	⌊B	3.14	4	3	-3.14	-3	-4	Plafond	⌈	Maximum	$3 \lceil 7 \leftrightarrow 7$	
B	⌈B	⌊B												
3.14	4	3												
-3.14	-3	-4												
	Plancher	⌊	Minimum	$3 \lfloor 7 \leftrightarrow 3$										
$*B \leftrightarrow (2.71828...) * B$	Exponentielle	*	Puissance	$2 * 3 \leftrightarrow 8$										
$\odot * N \leftrightarrow N \leftrightarrow * \odot N$	Logarithme naturel	⊙	Logarithme	$A \odot B \leftrightarrow \text{Log } B \text{ base } A$ $A \odot B \leftrightarrow (\odot B) \div \odot A$										
$ - 3.14 \leftrightarrow 3.14$	Valeur absolue		Résidu	Cas	A B									
				A≠0	B-(A)×⌊B÷ A									
				A=0,B≥0	B									
				A=0,B<0	Erreur de domaine									
$!0 \leftrightarrow 1$ $!B \leftrightarrow B \times !B - 1$ ou $!B \leftrightarrow \text{gamma}(B+1)$	Factorielle	!	Coef. binomial	$A!B \leftrightarrow (!B) \div (!A) \times !B - A!$ $2!5 \leftrightarrow 10 \quad 3!5 \leftrightarrow 10$										
$?B \leftrightarrow \text{choix aléatoire à partir de } !B$	Tirage	?	Donne	Fonction Mixte (Voir Tableau 3.1)										
$OB \leftrightarrow B \times 3.14159...$	Multipliat. par Pi	o	Circulaire	Voir tableau à gauche										
$\sim 1 \leftrightarrow 0 \quad \sim 0 \leftrightarrow 1$	Non	~												
			^ ET	A	B	A^B								
			v OU	0	0	0								
			* NONET	0	1	0								
			v NONOU	1	0	0								
				1	1	0								
			< Inférieur	Relations										
			≤ Inf. ou égal	Le résultat est égal à 1 si la relation est vérifiée,										
			= Egal	et à 0 si elle ne l'est pas:										
			≥ Sup ou égal											
			> Supérieur	$3 \leq 7 \leftrightarrow 1$										
			≠ Différent	$7 \leq 3 \leftrightarrow 0$										

(-A) OB	A	A OB
$(1-B \times 2) * .5$	0	$(1-B \times 2) * .5$
Arcsin B	1	Sinus B
Arccos B	2	Cosinus B
Arctan B	3	Tangente B
$(-1+B \times 2) * .5$	4	$(1+B \times 2) * .5$
Arcsinh B	5	Sinh B
Arccosh B	6	Cosh B
Arctanh B	7	Tanh B

Tableau des Fonctions Circulaires dyadiques

Tableau 3.1: Fonctions primitives scalaires

Nom	Forme ¹	Définition ou exemple ²
Dimension	ρA	$\rho P \leftrightarrow 4$ $\rho E \leftrightarrow 3\ 4$ $\rho 5 \leftrightarrow 1\ 0$
Restructuration	$V\rho A$	Restructure A à la dimension V $3\ 4\rho 1\ 2 \leftrightarrow E$
linéarisation	$,A$	$12\rho E \leftrightarrow 1\ 12$ $0\rho E \leftrightarrow 1\ 0$ $,A \leftrightarrow (\times/\rho A)\rho A$ $,E \leftrightarrow 1\ 12$ $\rho,5 \leftrightarrow 1$
Chainage	V,V	$P,1\ 2 \leftrightarrow 2\ 3\ 5\ 7\ 1\ 2$ $'T','HIS' \leftrightarrow 'THIS'$
Indice ^{3\ 4}	$V[A]$ $M[A;A]$ $A[A;...]$ $...;A]$	$P[2] \leftrightarrow 3$ $P[4\ 3\ 2\ 1] \leftrightarrow 7\ 5\ 3\ 2$ $E[1\ 3;3\ 2\ 1] \leftrightarrow 3\ 2\ 1$ 11 10 9 $E[1;] \leftrightarrow 1\ 2\ 3\ 4$ $ABCD$ $E[;1] \leftrightarrow 1\ 5\ 9$ $'ABCDEFGHijkl'[E] \leftrightarrow EFGH$ IJKL
Générateur d'indice ³	$1S$	S premiers entiers ¹ $14 \leftrightarrow 1\ 2\ 3\ 4$ $10 \leftrightarrow$ un vecteur vide
Indice de ³	$V1A$	premier indice de A dans V , ou $1+\rho V$ $P13 \leftrightarrow 2$ $5\ 1\ 2\ 5$ $P1E \leftrightarrow 3\ 5\ 4\ 5$ $4\ 4\ 1\ 4 \leftrightarrow 1$ $5\ 5\ 5\ 5$
Extraction	$V+A$	Extrait (supprime) $ V[I]$ premiers éléments de la coordonnée I . (les derniers si $V[I]<0$) $2\ 3+X \leftrightarrow ABC$ EFG
Suppression	$V+A$	$-2+P \leftrightarrow 5\ 7$
Tri ⁵ croissant	ΔA	la permutation qui ordonne $\Delta 3\ 5\ 3\ 2 \leftrightarrow 4\ 1\ 3\ 2$
Tri ⁵ décroissant	∇A	A (ordre croissant ou décroissant) $\nabla 3\ 5\ 3\ 2 \leftrightarrow 2\ 1\ 3\ 4$
Compression ⁵	V/A	$1\ 0\ 1\ 0/P \leftrightarrow 2\ 5$ $1\ 0\ 1\ 0/E \leftrightarrow 5\ 7$ 9 11 $1\ 0\ 1/[1]E \leftrightarrow 1\ 2\ 3\ 4 \leftrightarrow 1\ 0\ 1/E$ 9 10 11 12
Expansion ⁵	$V\backslash A$	$1\ 0\ 1\backslash 12 \leftrightarrow 1\ 0\ 2$ $1\ 0\ 1\ 1\ 1\backslash X \leftrightarrow EFGH$ I JKL $A\ BCD$
Symétrie	ϕA	$DCBA$ $IJKL$ $\phi X \leftrightarrow HGFE$ $\phi[1]X \leftrightarrow eX \leftrightarrow EFGH$ $LKJI$ $\phi P \leftrightarrow 7\ 5\ 3\ 2$ $ABCD$
Rotation ⁵	$A\phi A$	$3\phi P \leftrightarrow 7\ 2\ 3\ 5 \leftrightarrow -1\phi P$ $1\ 0\ -1\phi X \leftrightarrow EFGH$ LIJK $BCDA$
Transposition	$V\phi A$	la coordonnée I de A devient la coordonnée $V[I]$ du résultat $2\ 1\phi X \leftrightarrow BFJ$ CGK $1\ 1\phi E \leftrightarrow 1\ 6\ 11$ DHL
	ϕA	Transpose les 2 dernières coordonnées $\phi E \leftrightarrow 2\ 1\phi E$
Appartenance	$A \in A$	$\rho W \in Y \leftrightarrow \rho W$ $E \in P \leftrightarrow 1\ 0\ 1\ 0$ $P \in 14 \leftrightarrow 1\ 1\ 0\ 0$ $0\ 0\ 0\ 0$
Décodage	$V1V$	$1011\ 7\ 7\ 6 \leftrightarrow 1776$ $24\ 60\ 6011\ 2\ 3 \leftrightarrow 3723$
Codage	$V1S$	$24\ 60\ 6013723 \leftrightarrow 1\ 2\ 3$ $60\ 6013723 \leftrightarrow 2\ 3$
Donne ³	$S?S$	donne aléatoire de W éléments de $1Y$

Tableau 3.1 : Fonctions primitives mixtes

1. Les restrictions au rang des arguments sont indiquées par: S pour un scalaire, V pour vecteur, M pour matrice et A pour n'importe laquelle de ces variables. A l'exception du premier argument de $S;A$ ou $S[A]$, un scalaire peut remplacer un vecteur. Un tableau à un élément peut remplacer une variable scalaire.

2. Tableaux utilisés

dans les exemples: $P \leftrightarrow 2 \ 3 \ 5 \ 7$ $E \leftrightarrow \begin{matrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{matrix}$ $X \leftrightarrow \begin{matrix} ABCD \\ EFGH \\ IJKL \end{matrix}$

3. La fonction dépend de l'origine des indices.

4. L'élision d'un indice quelconque sélectionne tout ce qui se trouve le long de cette coordonnée.

5. La fonction s'applique le long de la dernière coordonnée; les symboles $/$, $\backslash$, et Θ équivalent à $/$, $\backslash$, et ϕ respectivement, mais la fonction s'applique le long de la première coordonnée. Si $[S]$ apparaît après l'un quelconque des symboles, la coordonnée y relative est déterminée par la variable scalaire S .

Notes sur le tableau 3.1

Nous rappelons donc Tables 3.1. les opérateurs de base du langage que nous utiliserons dans la description fonctionnelle.

Lorsqu'une transformation utilisée fréquemment pour décrire le comportement d'un système ne s'exprime pas directement par un opérateur de base du langage, il est plus simple de la déclarer comme fonction. Ce qui revient à l'inclure comme opérateur spécifique de cette description. Prenons à titre d'exemple la fonction comptage. Elle se traduit à l'aide des opérateurs de base A.P.L. par la fonction à un seul argument suivante :

$$\nabla B \leftarrow CPT \ A$$

$$B \leftarrow ((PA) P2) T (((PA) P2) \perp A) + 1 \nabla$$

La fonction CPT est ensuite incorporée comme transformation primitive de la description.

3.2.4. Evolution parallèle et description série

La modification d'un identificateur est donc décrite à un instant donné par une instruction d'affectation écrite en langage A.P.L.

Si l'on considère une séquence d'entrée particulière, elle se traduit par un ensemble de modifications sur différents identificateurs. Ces modifications ne sont pas en général totalement ordonnées dans le temps puisque, dans un circuit logique, différents sous-systèmes peuvent évoluer de manière totalement asynchrone (voir la relation d'ordre définie au paragraphe 3.2.2.). Si l'on désire décrire cette réalité du comportement, il est donc nécessaire de développer un système permettant de prendre en compte les notions de parallélisme (COU 1), (CAR 1).

En fait, les descriptions d'événements parallèles conduisent à des langages complexes et de toute façon difficiles à utiliser pour la génération des tests.

Nous avons donc choisi de simplifier le modèle en décrivant l'ensemble des modifications séquentiellement. Cette transformation de l'ensemble des modifications en suite totalement ordonnée doit être telle qu'elle ne change aucun des résultats fonctionnels.

Cette simplification de la description n'est pas en fait très contraignante. Dans un certain sens, elle est moins grossière que l'approximation faite lorsque l'on modélise un circuit logique à l'aide des fonctions booléennes ou du logigramme ne prenant pas en compte les temps de propagation. Dans ce modèle, la notion de temps est approchée par la relation d'ordre.

Montrons qu'il est toujours possible d'obtenir une description série qui ne modifie aucun des résultats fonctionnels. Considérons pour cela l'application d'une séquence d'entrée appartenant au fonctionnement normal. Elle se traduit comme nous l'avons vu précédemment, par un ensemble de modifications d'identificateurs décrites en langage A.P.L. Traçons un graphe dans lequel chaque modification est représentée par un sommet et tel qu'il existe un arc (i,j) si les modifications i et j sont ordonnées avec $i < j$ (relation d'ordre définie paragraphe 3.2.2.). On obtient alors un graphe d'ordre partiel dont un exemple est donné figure 3.1. Prenons deux sommets i et j non ordonnés de ce graphe. Les modifications correspondantes s'effectuent de manière asynchrone. Donc, si l'une de ces modifications, soit i , change la valeur contenue dans un identificateur M (M apparaît dans la partie gauche de l'instruction i) il est impossible que l'instruction j contienne l'identificateur M si le système décrit est déterministe. Cela nous permet d'énoncer la propriété suivante :

Propriété 3.2. Pour un système déterministe, on ne change pas les résultats fonctionnels en effectuant un ordonnancement des modifications. Figure 3.2.

Remarque : Dans certains cas, cet ordonnancement peut amener à introduire de nouveaux identificateurs qui n'ont aucune réalité physique.

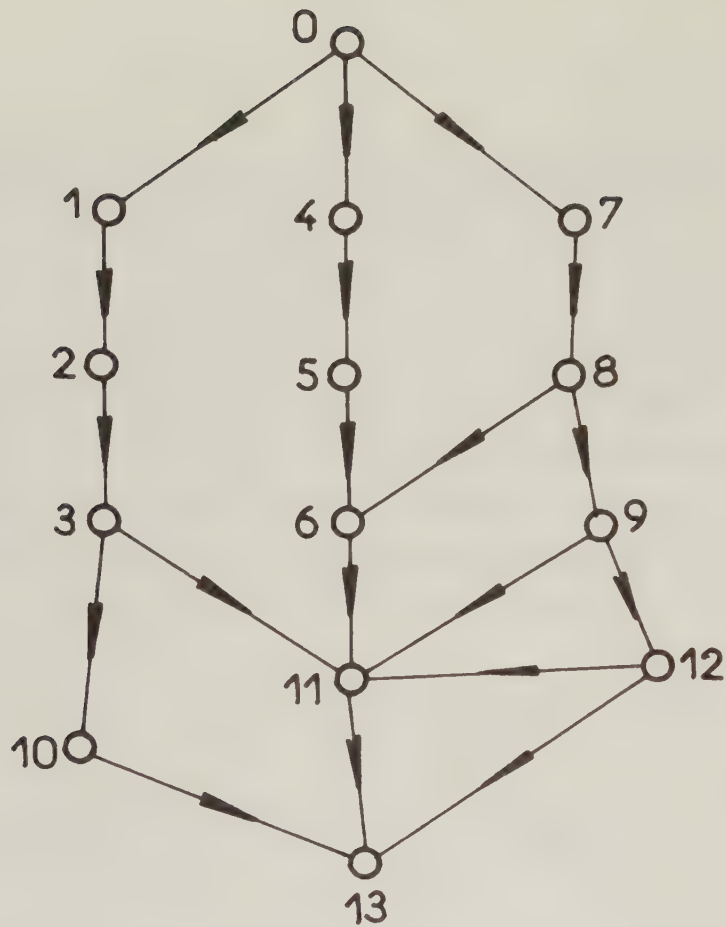


Figure 3.1. Graphe d'ordre partiel d'un ensemble de modifications.

$$(0) \longrightarrow (1, 4, 7) \longrightarrow (2, 5, 8) \longrightarrow (3, 6, 9) \longrightarrow (10, 11, 12) \longrightarrow (13)$$

Figure 3.2. Un ensemble d'ordonnancements possibles.

3.2.5. Description d'un sous-système : ensemble fermé d'identificateurs et contraintes

Définition 3.2. Un ensemble d'identificateurs d'un système logique est un ensemble fermé si toute modification de valeur d'un élément de cet ensemble, due à une séquence appartenant au fonctionnement normal, s'exprime uniquement à l'aide d'autres éléments de cet ensemble.

Dans ce cas particulier, il est possible d'effectuer une description fonctionnelle du sous-système composé de cet ensemble d'identificateurs. Un exemple simple d'un ensemble fermé d'identificateurs est la partie traitement des données d'un système dont l'évolution est commandée par un séquenceur.

Cette définition montre bien qu'il n'est toutefois pas possible d'utiliser une telle description pour modéliser une partie quelconque d'un système logique. Cette définition est constructive en ce sens qu'elle indique une méthodologie pour obtenir l'ensemble des identificateurs nécessaires à une description.

La partition du système en deux sous-ensembles, la partie décrite et son complément, conduit à introduire un nouvel ensemble de variables qui effectue le lien entre les deux sous-systèmes.

Comme nous l'avons postulé précédemment, ces variables n'interviennent pas de façon explicite dans l'expression des modifications. Elles ont donc soit une valeur fixe (indépendante des opérandes de la modification) soit une structure fixe (par exemple, une impulsion d'horloge). Pour cette raison, nous les nommerons contraintes $\mathcal{C}$.

Les contraintes ont pour rôle de permettre l'élaboration d'une modification ou encore le passage d'une modification à la suivante. Dans l'exemple particulier précédent, d'une partie traitement des données, ces contraintes correspondent aux variables de commande provenant du séquenceur.

3.2.6. Mise en séquence des modifications

Séquences d'initialisation

Lorsque l'on met un système logique sous tension, son état initial est totalement inconnu. Il est théoriquement toujours possible de déterminer le comportement de cette machine à l'aide de séquences de positionnement dans un état final. Elles peuvent être obtenues sans autre information que le logigramme, la table de fluence ou les fonctions logiques associées au circuit (RIC 2). La recherche de ces séquences n'est en général pas simple sauf pour les circuits de taille triviale.

En pratique, le concepteur d'un système logique complexe prévoit un ensemble de séquences d'entrée permettant de positionner le système dans un état ou un ensemble d'états S , de manière que le comportement du système soit parfaitement déterminé pour toute séquence d'entrée, appartenant au fonctionnement normal, appliquée à partir de S . Négliger l'existence de ces séquences est un des graves défauts des descriptions par table de fluence, logigramme ou fonctions logiques associées au circuit.

Ces séquences d'initialisation constitueront le début de la description fonctionnelle.

Séquences de modifications : actions conditionnelles

Comme nous l'avons vu au paragraphe 3.2.5., il est toujours possible de décrire l'ensemble des modifications dues à toute séquence de fonctionnement normal par une procédure série qui ne change aucun des résultats fonctionnels. Cela suggère l'utilisation d'un organigramme pour décrire l'enchaînement des modifications.

Il s'agit d'une des différences essentielles entre la description au niveau portes élémentaires et celle au niveau transferts entre

registres (T.R.). Comme il est souligné dans (BAR 1) les descriptions par transferts entre registres considèrent que les éléments passent par des périodes d'activité et de repos. Bien sûr, en réalité, tous les éléments sont actifs en permanence et l'abstraction consiste à ne préciser que les activités intéressantes.

Cette abstraction se traduit par l'utilisation d'instructions de branchement qui spécifient la modification suivante qu'il sera réalisée. De même, que pour les instructions d'affectation, les instructions conditionnelles seront écrites en langage A.P.L. Pour rester cohérent avec le début de notre exposé, il s'agira donc d'instruction du type affectation-branchement puisque le choix de l'activité suivante est prise sur la valeur d'un identificateur éventuellement modifié.

3.2.7. Liens avec la partie structure de réalisation

La partie fonctionnelle de la description est directement liée à la réalisation technologique. En fait, un modèle de la réalisation est un des éléments nécessaires de la description (paragraphe 3.1.).

La correspondance entre la partie réalisation et la partie mission est réalisée à l'aide des ensembles d'identificateurs et de contrainte attachés à chacune des deux parties. Nous avons choisi pour matérialiser cette correspondance une bijection entre les deux ensembles de variables. Ainsi, un objet identifié dans la réalisation physique se retrouvera avec la même identité dans la partie mission.

Une modification ou plutôt l'instruction qui lui correspond permet alors de définir le circuit combinatoire extrait de la réalisation qui est le système physique produisant cette modification.

Les entrées de ce circuit combinatoire sont les identificateurs à droite de l'instruction et les variables de contrainte. Les sorties sont les identificateurs à gauche de l'instruction. Les variables de

contrainte sont les entrées de ce circuit qui portent une valeur booléenne déterminée, ou une forme de signal déterminé.

3.3. DEFINITION FORMELLE D'UNE DESCRIPTION MISSION-REALISATION

Définitions 3.3.

a. Une description Mission-Réalisation (D.M.R.) est donc un triplet

$$D M R \triangleq (D M_0, D R_0, A)$$

où $D M_0$ est la description de la mission avec son interprétation

$D R_0$ est la description de la réalisation avec son interprétation

A est une correspondance entre $D M_0$ et $D R_0$.

b. Description Mission $D M$ et son interprétation

b_1 . La description de la Mission $D M$ est un quintuple

$$D M \triangleq (I, C, ME, \mathcal{V}, e_K) \text{ où}$$

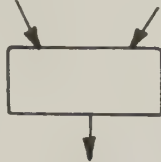
* $I = \{i\}$ est un ensemble d'identificateurs dont on précise pour chacun d'eux les dimensions. I peut être décomposé en quatre ensembles disjoints $I = X \cup Z \cup Z_L \cup M$; X sont les entrées, Z les sorties primaires, Z_L les sorties de lien, M les mémoires.

* $C = \{c\}$ est un ensemble de variables de contrainte dont on précise pour chaque élément les dimensions.

* $ME = \{me\}$ est un ensemble de modifications ou missions élémentaires. Chaque élément me est donné sous la forme d'un assignement $i \leftarrow f(I)$ où i appartient à I . f est une expression algébrique écrite dans un langage L .

* γ est une application de $ME' = ME \cup \text{DEBUT} \cup \text{FIN}$ dans lui-même,
 (γ, ME') est un graphe comportant quatre types de noeuds :

- le noeud séquence : qui a n arcs intérieurs et 1 arc extérieur, symbolisé par



- le noeud de branchement : qui a n arcs intérieurs et $p > 1$ arcs extérieurs. Il est symbolisé par



- le noeud début : qui a zéro arc intérieur et un arc extérieur. Il est symbolisé par



- le noeud fin : qui a n arcs intérieurs et aucun arc extérieur. Il est symbolisé par



* $e_K = \{ e_k \}$ est un ensemble d'expressions booléennes (éventuellement l'expression 0 vide) définies sur l'ensemble C . Une expression e_k est associée à chaque arc et chaque noeud du graphe (ME', γ)

b_2 . Une interprétation $D M_0$ de $D M$ est un quintuple

$$D M_0 \triangleq (D M, \mathcal{D}_X, \mathcal{D}_C, f_K, r)$$

où

* $D M$ est une description de la mission

* $\mathcal{D}_X$ est le domaine des variables X

* $\mathcal{D}_C$ est le domaine des variables C

* $f_K = \{f_k\}$ est un ensemble de fonctions booléennes simples qui à tout e_k fait correspondre sa valeur pour un élément de $\mathcal{D}_C$. $\mathcal{D}_C \xrightarrow{f_k} \{0,1\}$

* r est l'ensemble des règles d'évolution dans D M.

- le noeud début est actif, tous les autres sont inactifs.

- un seul noeud ou arc est actif à la fois.

- dès que la fonction contrainte associée à un noeud est égale à 1, l'expression me (modification élémentaire) attachée à ce noeud est évaluée.

- suivant la valeur calculée, l'un des arcs extérieurs est activé lorsque sa fonction contrainte est satisfaite. L'activation, désactive le noeud source.

- lorsque le noeud fin est désactivé, on active de nouveau le noeud début.

Une interprétation DM_0 est cohérente si pour tout noeud de branchement l'expression associée peut prendre un ensemble de valeurs en nombre inférieur ou égal au nombre d'arcs extérieurs. Elle est vivante si pour tout arc et tout noeud du graphe (ME', γ) il existe au moins une séquence d'entrée appartenant à $(\mathcal{D}_X \times \mathcal{D}_C)^*$ et permettant de l'activer. Nous supposons toujours que les descriptions considérées sont cohérentes et vivantes.

c. Description de la réalisation D R

c₁. Une description de la réalisation D R est un quintuple

$$D R \triangleq (J, E_J, S_J, l, T_J)$$

où

* $J = \{j\}$ est un ensemble d'identificateurs dans lequel on peut en particulier retrouver les entrées EN et les sorties SO du système.

* $E_J = \{E_j \mid E_j \text{ est l'ensemble des entrées de l'identificateur } j\}$

* $S_J = \{S_j \mid S_j \text{ est l'ensemble des sorties de l'identificateur } j\}$

* $T_J = \{T_j \mid T_j \text{ est une correspondance entrées-sorties de } j\}$

T_j peut être donnée sous la forme d'une table, d'un graphe, d'une fonction booléenne, ou d'une description de Mission.

* l est une application de l'ensemble des sorties dans l'ensemble des entrées auquel on ajoute l'élément vide λ

$$l : \begin{matrix} S_j \\ J \end{matrix} \longrightarrow \begin{matrix} U \\ J \end{matrix} \begin{matrix} E_j \\ \cup \end{matrix} \lambda$$

c₂. Une interprétation D R₀ de D R est un triplet

$$D R_0 = (D R, \mathcal{D}_{EN}, rg)$$

où

* D R est la description de la réalisation

* $\mathcal{D}_{EN}$ le domaine des entrées primaires

* rg l'ensemble des règles qui, étant donnée une séquence d'entrée de $\mathcal{D}_{EN}^*$ permettent de calculer la réponse du système à l'aide de la correspondance entrées-sorties de chaque élément.

d. A correspondance entre $D M_0$ et $D R_0$

A est une correspondance biunivoque entre I et un sous-ensemble I' de J

$$I \xrightarrow{A} I' \subseteq J$$

de telle sorte que pour toute séquence appartenant à $(\mathcal{D}_X \times \mathcal{D}_C)^*$ la suite des modifications des valeurs des éléments de I soit un ordonnancement de l'ensemble des valeurs prises par les éléments correspondants de I'.

3.4. EXEMPLE D'ILLUSTRATION

Nous traiterons dans ce paragraphe un exemple très simple. Le circuit a été choisi dans le but d'illustrer les définitions données précédemment. Sa petite taille ne permet pas toutefois de dégager clairement tous les avantages du mode de description, qui sont apparus bien plus à propos de l'exemple pratique (RIC 1), (SOU 1) traité en annexe. La dimension de ce dernier est plus de 100 fois celle de l'exemple traité ci-dessous.

3.4.1. Description de la réalisation

Les figures 3.3. et 3.4. sont deux descriptions de la réalisation. Elles sont à des niveaux de détail différents. Sur la figure 3.4. sont identifiés les éléments qui sont communs aux descriptions Mission et Réalisation. Les ensembles d'entrées et de sorties (E_j et S_j) de chaque identificateur, ainsi que les connexions entre identificateurs sont explicitement précisés par la figure 3.4. Les correspondances entrées et sorties (T_j) sont données pour une part (circuits combinatoires) par des logigrammes

les bascules d'autre part peuvent être décrites par les tables ou les graphes de fluence ou parfois par les logigrammes. Les entrées (E_N) sont : A1, B1, B2, RAZC, RAZM et appartiennent à l'ensemble $\{0,1\}$. Ce sont des niveaux. H est l'horloge et agit sur le front montant.

La liste des identificateurs est donc (figure 3.4.) :

- identificateurs scalaires : A1, S, TEST 1, TEST 2, M, RAZC, RAZM, B1, B2, H
- identificateurs vectoriels : R1 (1:8), R2 (1:8), C(1:4)

Parmi ces identificateurs :

A1, RAZC, RAZM, B1, B2, H sont des entrées

S, TEST 1, TEST 2 sont des sorties

C, M, R1, R2 sont des mémoires.

3.4.2. Description de la mission

Sauf dans les cas très simples, il est impossible de préciser la mission d'un circuit en s'appuyant uniquement sur sa réalisation.

Comme nous l'avons vu d'une manière générale, la mission peut être définie du point de vue des entrées primaires comme l'ensemble de toutes les séquences que le système est susceptible de se voir appliquer au cours de son existence, nous les avons appelées séquences de fonctionnement normal. L'ensemble de ces séquences est généralement beaucoup plus réduit que l'ensemble de toutes les séquences d'entrées possibles. Ce qui est important c'est que le fonctionnement normal peut toujours être, nous dirons même doit toujours être décrit, car il correspond au domaine prévu par le concepteur. Du point de vue de la vérification, il contiendra tous les tests pour les pannes pouvant affecter sa mission.

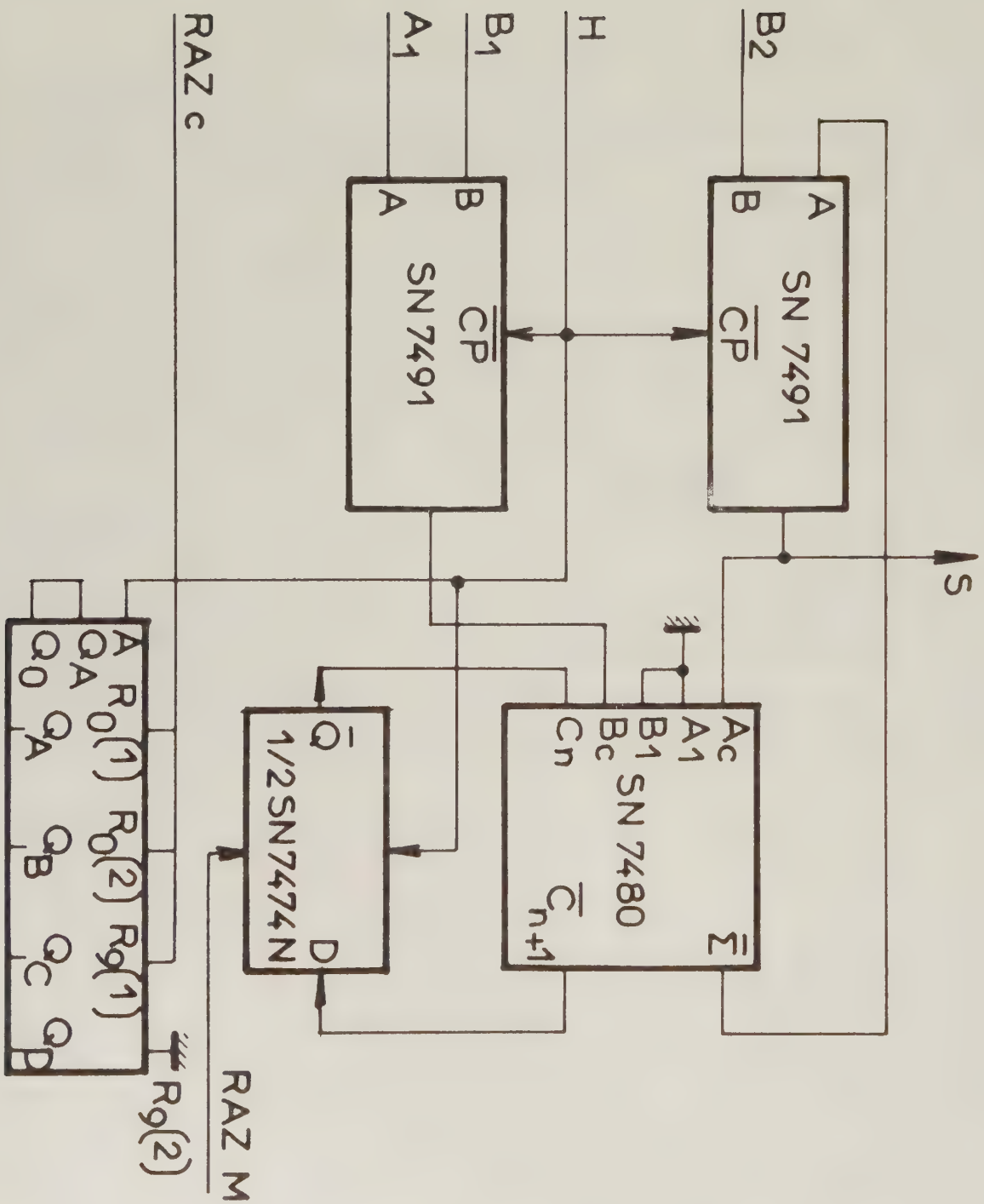


Figure 3.3. Exemple illustratif

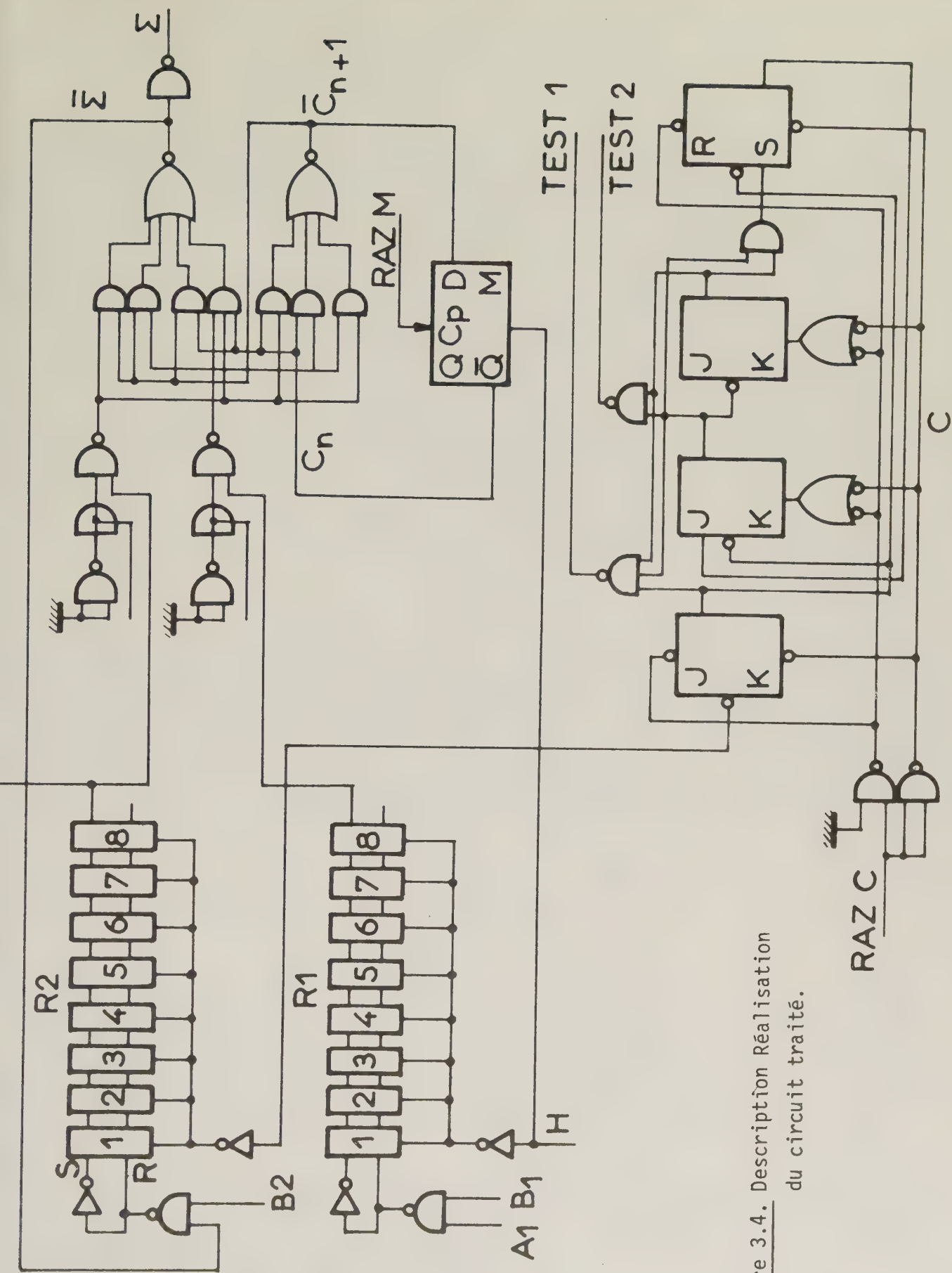
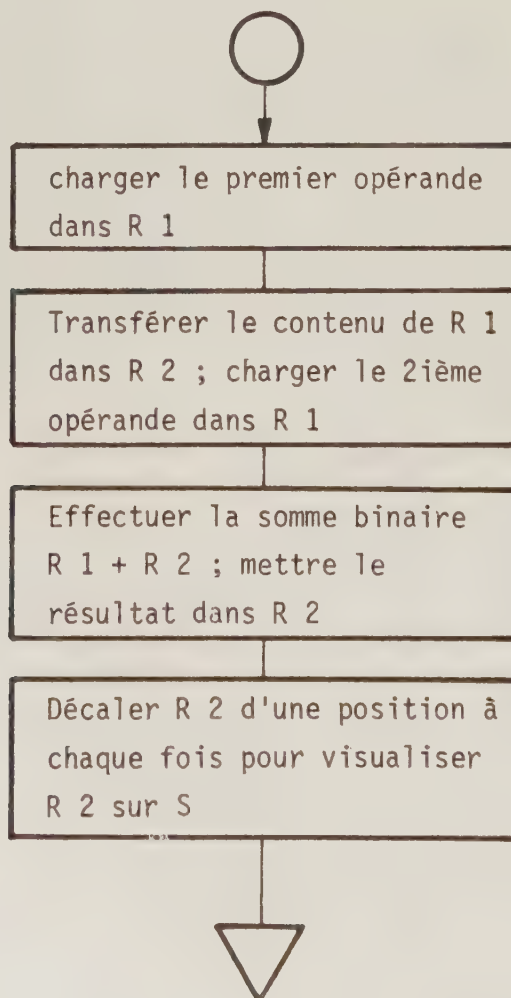


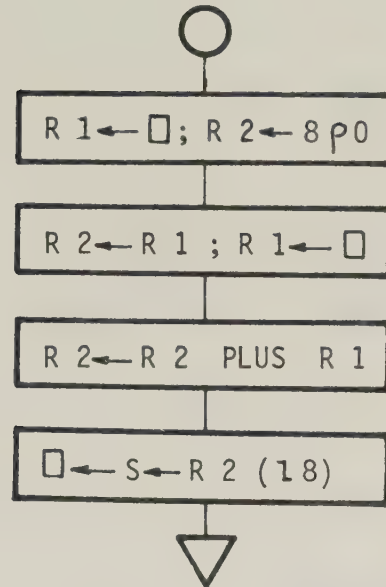
Figure 3.4. Description Réalisation
du circuit traité.

Il existe différents niveaux de description fonctionnelle. Nous décrirons plusieurs de ces niveaux pour bien préciser celui que nous avons défini dans le paragraphe 3.2.

a. La mission du circuit traité est : l'addition série de deux nombres de 8 bits chacun et dont la valeur est inférieure à 2^8 . Cette information n'est pas négligeable, mais elle ne peut conduire qu'à vérifier les 2^{14} opérations possibles. Si l'on désire réduire le nombre des opérations à vérifier, il faut décrire cette même mission à un niveau de détail plus grand.

b. Voici une décomposition de cette mission en quatre étapes, tout d'abord donnée sous la forme d'un organigramme informel puis avec des instructions A.P.L.





Cette description de la mission contient déjà plus d'information que la précédente, mais elle ne permet pas une définition des pannes au niveau des connexions internes. Ainsi, la meilleure solution, à notre avis, est de décrire la mission en s'appuyant sur la structure de réalisation. En particulier, cela permet une définition précise des pannes et une possibilité de retour à la structure pour l'automatisation de certaines tâches.

c. Prenons comme identificateurs de la description, ceux que nous avons déjà répertoriés sur la figure 3.4.

On remarque que si l'on choisit simplement comme variables A1, B1, B2, R1, R2, M, RAZM, il n'est pas possible d'expliquer le fonctionnement ; c'est parce qu'il ne s'agit pas d'un ensemble fermé d'identificateurs. Il faut rajouter le compteur c ainsi que les sorties TEST 1 et TEST 2, dont on comprend l'utilité sur la description de la mission, figure 3.6.

La figure 3.6. est donc une description de la mission telle que, pour toute séquence appartenant au fonctionnement normal, la suite des modifications des valeurs des éléments de ses identificateurs constitue un ordonnancement de l'ensemble des valeurs prises par les éléments correspondants de la structure de réalisation, figure 3.4. Les deux descriptions réunies sont donc ce que nous avons défini comme une D.M.R.

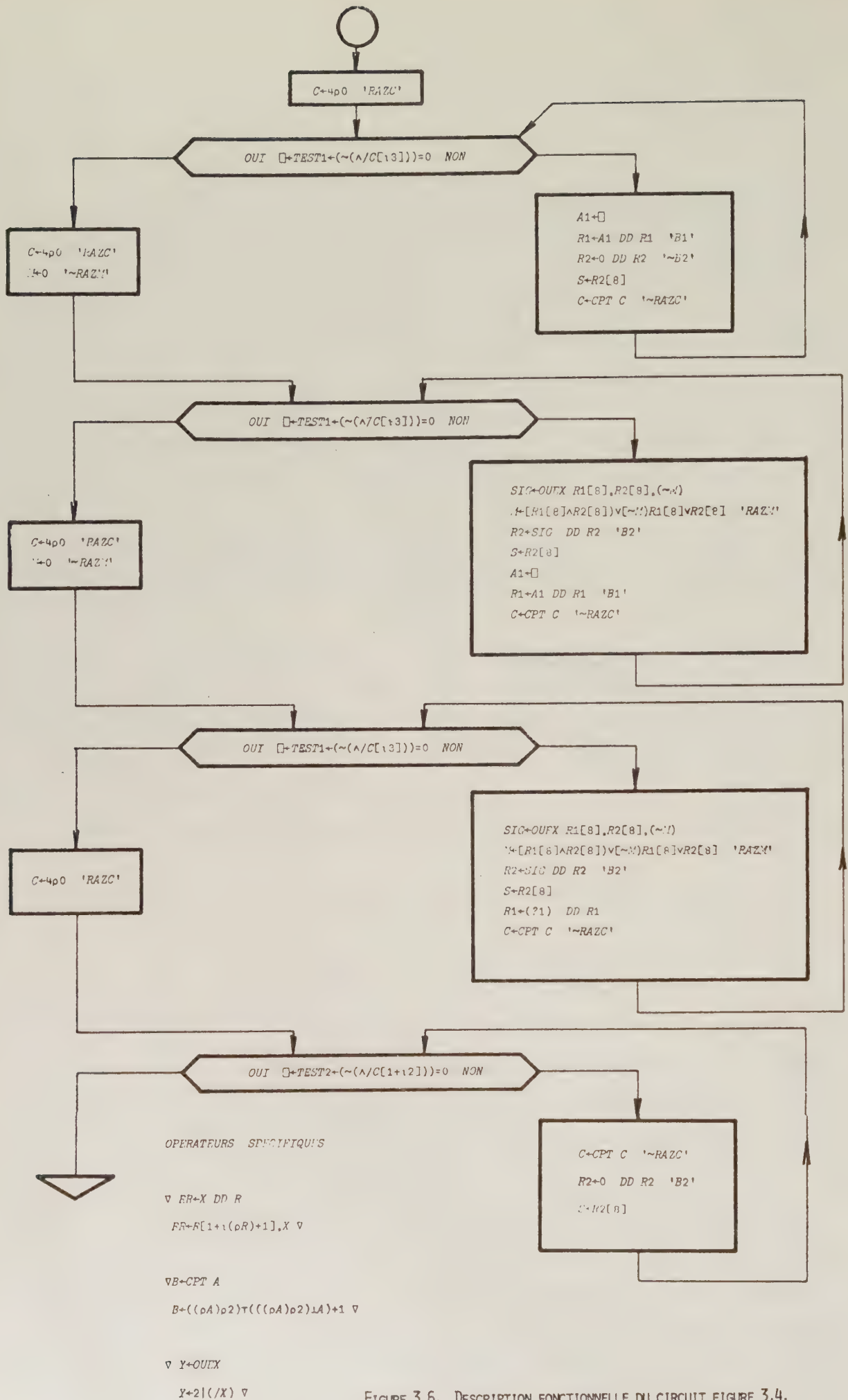


FIGURE 3.6. DESCRIPTION FONCTIONNELLE DU CIRCUIT FIGURE 3.4.

L'ensemble des variables a été découpé en deux parties :
les identificateurs et les variables de contrainte.

Les identificateurs sont : A_1 (entrée primaire)

S (sortie primaire)

TEST 1

et TEST 2 (sorties de lien)

R_1, R_2, C, M (mémoires)

Les variables de contrainte sont : $RAZC, RAZM, B_1$ et B_2 .

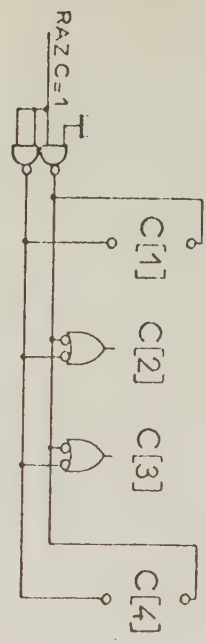
On remarque de plus la variable scalaire Σ qui est ici
introduite par la transformation de l'évolution parallèle en description
série.

3.4.3. Mise en évidence de la correspondance Mission-Réalisation

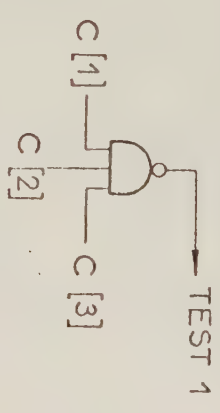
Aux paragraphes 3.2. et 3.3., nous avons introduit et souligné,
l'importance de la correspondance entre les deux parties de la description
Mission -Réalisation. Nous allons montrer sur cet exemple qu'à chaque
instruction correspond un circuit combinatoire extrait de la réalisation.
Il peut être obtenu de la manière suivante :

La partie gauche de l'instruction indique les sorties du
circuit. Les identificateurs de la partie droite et les variables de con-
trainte associées à l'instruction sont des entrées. On détermine donc le
circuit compris entre ces deux exemples d'identificateurs. Deux cas peuvent
se présenter : ou les entrées sont toutes des entrées identifiées ou cer-
taines ne le sont pas. Dans le premier cas, la construction est terminée,
dans le second on la poursuit jusqu'à se ramener au cas précédent.

(a) $C \leftarrow 4po (RAZ\ C)$



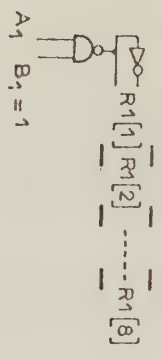
(b) $TEST\ 1 \leftarrow \sim (\wedge / C [1:3])$



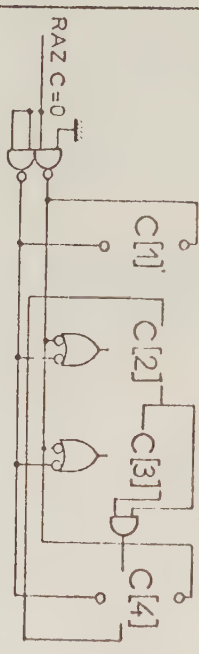
(c) $A1 \leftarrow \square$



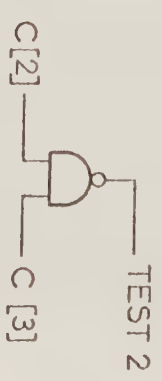
(d) $R1 \leftarrow A1\ DD\ R1\ (B1)$



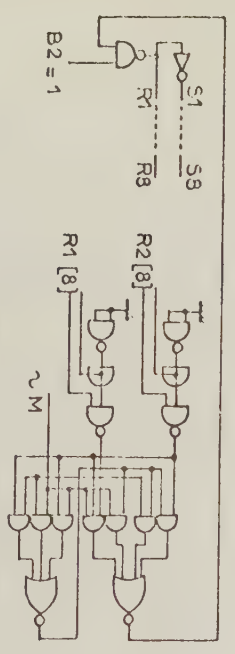
(e) $C \leftarrow CPT\ C\ (\sim RAZ\ C)$



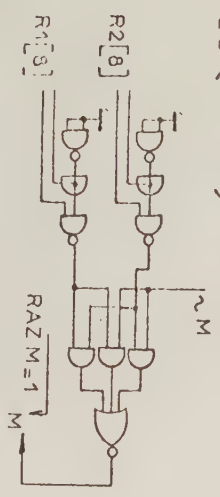
(f) $TEST\ 2 \leftarrow (\sim (\wedge / C [1+2]))$



(g) $\Sigma \leftarrow OUEX\ R1[8], R2[8], (\sim M)$



(h) $M \leftarrow (R1[8] \wedge R2[8]) \vee (\sim M) \wedge R2[8]$
 $\vee R1[8] (RAZ\ M)$



—EXAMPLES—

Pour les différentes instructions de la figure 3.4., on obtient les circuits combinatoires donnés ci-après sur la planche d'exemples.

Ainsi, chaque modification admet deux composantes. L'une est purement fonctionnelle : l'instruction ; l'autre purement structurelle : le circuit combinatoire correspondant à cette instruction. Si l'on suppose que chaque instruction est matérialisée par un circuit qui est technologiquement homogène on comprend mieux le rôle que jouera la description globale. Il faut souligner de plus l'importance de la partie fonctionnelle, qui, comme nous le verrons, peut permettre dans une certaine mesure indépendamment de la réalisation, d'aboutir à un ensemble de tests de vérification. Cela n'est pas négligeable si l'on se souvient que pour certains sous-systèmes fortement intégrés on ne connaît même plus la structure de réalisation avec précision.

3.5. UTILISATION DE LA DESCRIPTION POUR LE TEST

3.5.1. Modèle de panne

Le modèle de panne habituellement utilisé est le collage permanent d'une connexion appartenant au logigramme du circuit à vérifier. Il existe plusieurs raisons à cette hypothèse.

C'est tout d'abord par raison historique. En effet, les pannes de collage simple correspondaient certainement au type le plus probable pour les circuits conçus à l'aide de portes élémentaires. C'est ensuite en raison de la simplicité de leur traitement. Elles sont bien évidemment les plus faciles à traiter lorsque le modèle est un logigramme.

On a conservé ce modèle de panne pour sa simplicité de traitement sur les logigrammes, bien qu'il ne soit pas à l'évidence le plus probable lorsqu'il s'agit de circuits fortement intégrés.

En fait, si les ensembles de tests obtenus restent, dans une certaine mesure, satisfaisants c'est parce qu'ils détectent un ensemble de pannes bien plus conséquent que les pannes simples. Pour les circuits combinatoires, les études sur la représentation des pannes (SCH 1), (VER 1) ont montré que la détection des pannes simples impliquait la détection de nombreuses pannes multiples. Toutefois, la totalité des pannes détectées par un test est spécifique du test choisi. Aussi, ne peut-on juger de l'efficacité d'un ensemble de tests qu'à postériori.

Pour les circuits séquentiels, les propriétés d'implication ne s'étendent pas de manière générale et donc, même en ne considérant que les pannes simples, on est réduit à une mesure de l'efficacité d'un test à postériori.

L'hypothèse des pannes simples n'est donc motivée théoriquement pour les circuits séquentiels que par la simplicité de traitement.

Aussi, bien que la panne de collage unique d'une connexion puisse être traitée à l'aide de la D.M.R., nous avons choisi de définir un ensemble de pannes logiques permanentes plus simples à traiter qui doit aussi bien entraîner la détection d'un ensemble de pannes fortement probable.

Un premier type de panne de la description sera localisé au niveau d'une seule instruction. Cette panne est la transformation de l'instruction en une nouvelle instruction. Nous avons vu qu'à toute instruction correspond un circuit combinatoire contraint la panne sera considérée comme le collage d'une des connexions de ce circuit combinatoire.

Un deuxième type de panne sera la suppression d'un des arcs de l'organigramme de la description fonctionnelle.

3.5.2. Ensemble de conditions nécessaires

Le modèle de fonctionnement est évidemment une représentation simplifiée de la réalité. Il ne tient pas compte en particulier des retards introduits par chaque élément. Du point de vue purement fonctionnel, il donne toutefois une bonne image du comportement sans panne puisque ce circuit est la matérialisation de cette fonction plutôt que l'inverse.

En présence de certaines pannes, même de type collage permanent, qui conduisent le système au travers de séquences de fonctionnement non prévues lors de la conception, le modèle est souvent une représentation grossière du comportement et souvent ne le représente plus du tout.

Il semble alors hors de proportion de mettre en oeuvre des méthodes de génération très sophistiquées qui conduisent à des tests ayant une faible probabilité d'être acceptés dans la réalité.

Notre approche a donc été de rechercher un ensemble de conditions nécessaires pour qu'une séquence soit un test et de dégager celles qui avec certitude restent valables dans la réalité.

Condition 3.1. Une condition nécessaire pour qu'une séquence d'entrée soit une séquence de test est qu'elle implique le passage (l'activation) par le sommet de l'organigramme correspondant à l'instruction en panne s'il s'agit d'une panne associée à une instruction ou par l'arc en panne s'il s'agit d'une suppression d'arc.

Condition 3.2. Lorsque l'on applique une séquence de test :

• s'il s'agit d'une panne associée à une instruction, il est nécessaire que cette instruction fournisse une valeur différente lorsque la panne est présente ou non ;

* s'il s'agit d'une panne associée à une instruction de branchement, il est nécessaire que l'instruction suivante activée soit différente lorsque la panne est présente ou non.

Condition 3.3. Lorsque la panne s'est manifestée au niveau de l'instruction, il est ensuite nécessaire de la faire apparaître sur l'une au moins des sorties observables.

Il faut donc choisir la séquence d'entrée de manière que pour tout sommet s , activé en commun dans les organigrammes avec et sans panne, il existe au moins un identificateur portant une valeur différente parmi l'ensemble des identificateurs entrant dans le sous graphe successeur de s .

Ces trois conditions sont, bien évidemment, les trois étapes classiques de toutes les méthodes de test des pannes de collages permanents par propagation du défaut. Il existe toutefois des différences importantes :

- * les séquences permettant de réaliser les conditions 1 et 2 sont par construction des séquences appartenant au fonctionnement normal du circuit.
- * la panne peut se manifester non seulement par une différence de valeur d'un identificateur, mais aussi par une différence des deux chemins suivis dans les organigrammes avec et sans panne.

Comme nous l'avons souligné au début de ce paragraphe, le modèle est une bonne représentation du comportement essentiellement avant que la panne n'ait agi. Pour cette raison, nous pensons que les deux premières conditions sont bien plus impératives que la troisième.

Ainsi, contrairement aux méthodes classiques de test, nous commencerons par rechercher des séquences qui pour chaque arc et chaque instruction de l'organigramme remplissent les deux premières conditions.

3.5.3. Conception d'un ensemble de tests

Ensemble de pré-test

Les méthodes de recherche des tests consistent pour une bonne part en l'énumération de séquences qui réalisent un ensemble de conditions locales, par exemple, rendre possible la propagation d'une panne dans un logigramme ou encore, dans notre cas, passer par une instruction ou un arc.

Bien que les procédures d'énumération s'avèrent la solution la plus efficace sur calculateur, elles nécessitent tout de même des temps de calcul importants et croissant très rapidement avec la taille des systèmes traités. Aussi est-il toujours intéressant et souvent indispensable de pouvoir obtenir manuellement ce que nous appellerons un ensemble de prétest et qui est pratiquement un ensemble de séquences d'entrée permettant d'éliminer une bonne proportion des pannes par simulation directe de ces pannes et des séquences (FRI 1).

Il n'existe nulle part de définition de méthode ou de guide permettant d'obtenir de telles séquences. On est alors souvent réduit à une génération aléatoire lorsque l'on s'appuie uniquement sur la structure de réalisation. Par contre, si on analyse les deux premières conditions nécessaires, il est possible d'en tirer une règle simple pour obtenir une telle séquence en s'aidant de la partie fonctionnelle de la description.

Définition 3.3. Nous appellerons ensemble de pré-tests un ensemble de séquences d'entrée possédant les deux propriétés suivantes :

- tout arc de l'organigramme est parcouru au moins une fois,
- tout élément de chaque identificateur prend au moins une fois la valeur 1 et la valeur 0.

L'obtention d'une telle séquence ne présente, en général, pas de difficulté lorsqu'on s'aide de l'organigramme de la description fonctionnelle.

Il est toutefois fastidieux d'avoir à vérifier les deux propriétés précédentes pour chaque arc et chaque identificateur. A cette fin, on peut utiliser le programme A.P.L. correspondant à l'organigramme.

En effet, ce type d'information est disponible lorsqu'on effectue une trace des instructions pour un ensemble de données quelconque. Il suffit donc de prévoir la mise en mémoire des résultats sous forme de deux matrices : l'une correspondant aux arcs parcourus, l'autre aux valeurs prises par chaque élément des différents identificateurs.

A l'aide d'un programme de couverture des pannes, on recherche les pannes détectées par l'ensemble de pré-test. Il est ensuite nécessaire de rechercher les tests pour les pannes restantes.

Génération des tests pour une panne donnée

Considérons une instruction de l'organigramme de la description fonctionnelle. On peut lui associer un circuit combinatoire sous forme de structure ou sous forme de correspondance entrées-sorties. Ce circuit est obtenu à l'aide de la correspondance A entre D.M. et D.R. On possède d'autre part les contraintes, c'est-à-dire les entrées de ce circuit qui ont une valeur fixée au niveau de cette instruction.

Réalisation de la condition 2

Supposons une panne quelconque du circuit combinatoire associée à une instruction. Par hypothèse, elle est du type collage permanent. L'une des méthodes exposées aux chapitres 1 et 2 permet d'obtenir un test pour cette panne. Le fait que le circuit combinatoire ait des entrées fixées (contraintes) ne change rien aux procédures.

Ce test se présente donc sous la forme d'un vecteur dont les éléments ont l'une des trois valeurs $\{0, 1, X\}$. Il satisfait la condition 2 énoncée au paragraphe 4.4.2. Ce vecteur correspond en général à plusieurs

identificateurs, il doit donc être découpé en conséquence, et les valeurs affectées aux éléments des identificateurs concernés. Bien sûr, le calcul de ce vecteur nécessite un ensemble de choix à priori puisqu'il peut exister (voir chapitre 1 et 2) plusieurs vecteurs permettant de détecter cette même panne. Il faut donc mémoriser ces différents choix pour pouvoir éventuellement les remettre en question.

Pour faire une image du calcul qui doit être fait, on peut supposer qu'à tout identificateur correspond une pile. Le vecteur calculé précédemment constitue les premiers éléments d'un sous-ensemble de ces piles. Parmi celles initialisées, certaines correspondent à des entrées primaires, on leur affectera un numéro de couche permettant de retrouver l'ordre de la séquence.

Réalisation de la condition 1

Il s'agit ici de rechercher une séquence d'entrée permettant d'atteindre le sommet de l'organigramme correspondant à l'instruction en panne. Il faut de plus y aboutir avec des valeurs d'identificateurs compatibles avec celles trouvées précédemment.

Cela implique un double choix :

- celui du chemin correspondant à une telle séquence parmi tous les chemins possibles reliant le sommet début au sommet en panne de l'organigramme
- celui des valeurs des identificateurs le long de ce chemin.

Il est théoriquement possible d'envisager l'énumération de tous les cas partant du sommet en panne et remontant jusqu'au sommet début. Mais cette procédure, qui est celle des méthodes classiques sur les structures, est inexploitable pour des systèmes complexes. Heureusement, la description fonctionnelle permet, grâce à sa facilité de compréhension humaine, de choisir un des chemins. Nous supposons donc connu le chemin dans l'organigramme permettant de réaliser cette condition.

La procédure pour calculer les valeurs des identificateurs le long de ce chemin est alors simple. Elle consiste, étant donné des valeurs (0,1, X) pour les sorties, à calculer une combinaison d'entrée d'un circuit combinatoire permettant de justifier ces valeurs. Cela peut être mis en oeuvre soit à l'aide d'une simple relation d'entrée-sortie (par exemple l'expression $\sum \pi$ d'une fonction booléenne), soit à l'aide d'une structure (par exemple un logigramme ou un graphe de contact).

Cette procédure est appliquée pour chaque instruction du chemin choisi et les valeurs calculées sont placées sur les piles correspondantes en numérotant toujours les couches associées aux entrées primaires.

Si une incompatibilité est découverte, on retourne au dernier choix effectué, jusqu'à épuiser toutes les possibilités associées à ce chemin.

A la fin de cette étape, on dispose donc, dans les piles associées aux entrées primaires, d'une séquence d'entrée permettant de mettre en évidence la panne au niveau de l'instruction de départ.

Il faut ensuite pouvoir observer son effet sur les sorties primaires du système.

Réalisation de la condition 3

Soulignons une fois encore que cette dernière condition nous semble beaucoup moins importante que les deux précédentes.

La mise en évidence de la panne s'effectuera instruction par instruction. C'est-à-dire qu'après chaque nouvelle instruction, on s'assurera que le défaut est toujours représenté par une différence de valeur portée par un identificateur ou encore une différence de chemin entre les organigrammes avec et sans panne.

Deux cas peuvent se produire : l'instruction à traiter est commune ou non aux chemins parcourus dans l'organigramme avec ou sans panne.

- Si l'instruction est commune, on utilisera une méthode de propagation de la faute au travers du circuit combinatoire contraint correspondant à l'instruction.

- Par contre, si les chemins se séparent, on choisira les entrées de manière aléatoire, puisqu'il est peu probable que deux traitements différents conduisent à un même résultat. Afin de pouvoir éventuellement changer de choix les entrées non spécifiées du circuit, seront choisies dans l'ordre binaire naturel par exemple.

Cette procédure est appliquée jusqu'à atteindre des sorties primaires du système ou encore jusqu'à aboutir à un cycle.

L'organigramme de la figure 3.9. résume cette procédure d'obtention des tests.

3.6. CONCLUSIONS

La complexité de la vérification des systèmes logiques due, à sa nature fortement combinatoire, à la taille des systèmes impliqués, à la diversité des modèles associés au sous systèmes : fonctions booléennes, structures de réalisations, à la difficulté de compréhension de ces modèles, nous a conduit à définir un nouveau modèle constitué essentiellement de deux parties intimement liées :

La première s'appuie sur un langage évolué et traduit le fonctionnement normal sous la forme d'un organigramme.

La seconde est le circuit lui-même schématisé par une interconnexion d'éléments dont le fonctionnement est précisé par l'un des modèles classiques.

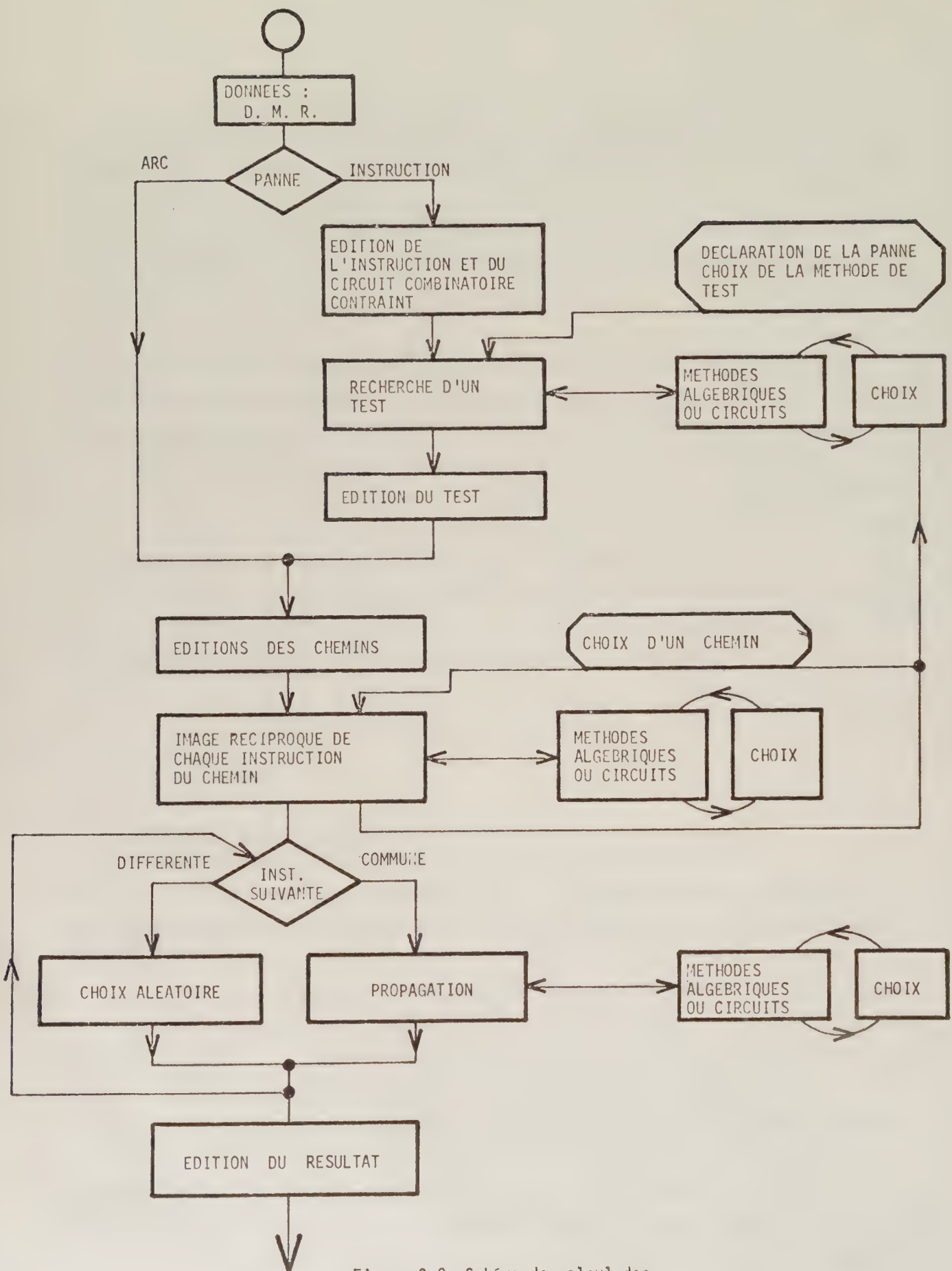


Figure 3.9. Schéma de calcul des séquences de vérification.

La correspondance entre ces deux paramètres est assurée par une même dénomination des identificateurs et des contraintes de l'organigramme et les entrées-sorties des éléments du circuit.

Le domaine de validité des modèles classiques se limite au fonctionnement normal et la complexité ne permet pas d'utiliser un modèle plus fin. Cela nous a conduit à remarquer que toute méthode de génération des tests pour un système logique complexe ne peut actuellement qu'être heuristique.

Nous avons donc mis en évidence les conditions qui nous ont semblé les plus fortes et qui correspondent en fait à la partie de la séquence et vérification que l'on peut choisir avec certitude parmi les séquences de fonctionnement normal. Ces règles simples se sont avérées très efficaces sur les exemples traités et nous ont permis de définir la notion d'ensemble de pré-test qui diminue considérablement le nombre des pannes à étudier. Enfin, nous avons montré la possibilité d'une génération automatique. Elle s'appuie, bien sûr, sur une méthode d'énumération implicite. Cette énumération est simplifiée par la présence des contraintes, la possibilité d'utiliser les différents modèles élémentaires à un niveau de hiérarchie inférieure et le choix d'un chemin appartenant au fonctionnement normal.

L'hypothèse de la panne localisée au niveau d'une instruction ne correspond à aucune réalité technologique. Elle ne peut trouver sa validation que dans les résultats obtenus. En fait, il est tout à fait possible de traiter dans le même esprit les pannes de collage classiques. Ce qui est remarquable c'est que les deux premières conditions nécessaires ne sont pas du tout modifiées. En effet, une telle panne se traduit au niveau de plusieurs instructions, il est donc nécessaire d'atteindre au moins l'une d'elles et de faire apparaître la panne sur les identificateurs de sortie. Seule, la mise en oeuvre de la condition 3 sera différente puisqu'il faut alors utiliser les modèles extraits de la structure en y insérant la panne considérée pour toutes les instructions concernées. Bien que théoriquement le modèle permette de décrire tout système logique, il est plus particulièrement adapté pour ceux à évolution en majorité série. La solution retenue

par (FAL 1) qui consiste en un ensemble de programmes partageant leurs variables, ainsi que le langage VDL/APL exposé dans (CAR 1) ne nous semblent pas être des solutions intéressantes pour la génération des tests. Un modèle de description composé de plusieurs organigrammes d'instructions écrites dans un langage évolué mais dont un seul à la fois serait validé paraît être la solution à retenir.

ANNEXE

EXEMPLE D'APPLICATION :

UNE PLAQUE MULTIPLICATION DIVISION

3A.1. DESCRIPTION MISSION-REALISATION DE LA PLAQUE

Nous avons appliqué ces notions à l'exemple pratique de la plaque multiplication division (SOU 1) du prototype ASMODEE 01 (LAP 1). La figure 3A.1. donne le schéma bloc de la plaque et symbolise la description de la réalisation.

Il s'agit d'un opérateur microprogrammé permettant d'effectuer la multiplication ou la division de deux nombres représentés en flottant par une mantisse de 16 bits plus signe et une caractéristique de 5 bits plus signe.

Les caractéristiques de la structure de réalisation sont :

- 47 connexions d'entrée-sortie qui doivent être connectées au bus d'ASMODEE
- 80 boîtiers de complexité différente
- une R.O.M. de $4 \times (32 \times 8)$ organisée en séquenceur de 32 états.

Le chiffre de 10.000 portes élémentaires permet d'avoir une bonne idée de la taille de cet opérateur et, dans l'état actuel des études sur le test des systèmes logiques, de le considérer comme un système complexe. Nous avons choisi de décrire avec précision la partie de cet opérateur correspondant au chemin des données, la partie commande étant réduite à sa plus simple expression.

La première étape de la description consiste à déterminer la liste des identificateurs du sous-système à décrire (figure 3A.2.).

Pour chaque identificateur, on a précisé la dimension par l'affectation du nombre de zéro correspondant. Dans le but de faciliter la compréhension et sans que cela soit indispensable, certaines sous-ensembles d'identificateurs ou fonctions d'autres identificateurs sont déclarés sous un

nom différent. Par exemple : $YR2 = R2(18)$, $RIR2 = R3(16,17)$; de même $PR1$ $PR2$, $PR1$ $PR3$ sont des fonctions logiques respectivement de $R1$, $R3$, $F4$ $G4$, $F5$ $G5$ et $R1$, $R2$, $F4$ $G4$, $F5$ $G5$.

Le fonctionnement est ensuite décrit sous la forme d'un organigramme pour lequel chaque bloc correspond le plus souvent à l'un des états du séquenceur (figure 3A.3.).

Les modifications effectuées à l'intérieur de chaque bloc ou état sont précisées par le "listing" A.P.L. 1130 (figure 3A.4.).

On remarque en particulier, pour les fonctions ETAT 0, ETAT 2, ETAT 3, la transformation d'un ensemble de modifications simultanées en actions séries. Toutes les modifications sont ici traduites uniquement à l'aide des opérateurs de base du langage A.P.L. 1130. Il est encore possible, pour augmenter la clarté, de définir un ensemble d'opérateurs spécifiques, figure 3A.5. On obtiendrait alors un nouveau "listing" A.P.L. en remplaçant les instructions concernées par ces fonctions.

3A.2. RECHERCHE DES TESTS

L'obtention et la validation des tests sont décrits en détail dans (CAB 1).

La vérification de l'ensemble des pannes détectées a été réalisée par une simulation physique à l'aide d'un testeur spécifique de ce type de plaque. On connaît les avantages et les inconvénients de ce type de démarche. Les pannes simulées sont bien sûr celles permises par la simulation physique. Leur nombre s'élève à 2.500. Il s'agit des collages des entrées et sorties des boîtiers, des courts-circuits entre pattes voisines, des inversions de boîtiers, des pannes d'alimentation.

Un ensemble de pré-test a été choisi manuellement. Il se composait de 12 opérations (multiplications et divisions). La simulation physique a permis de montrer que cet ensemble détectait 95 % des pannes considérées.

L'application hors ligne du programme DEDALE (TAB 1) a conduit à la détermination (CAV 1) de 3 opérations supplémentaires portant le taux de pannes détectées à 98,8 %. Les pannes non détectées ont été reconnues redondantes.

Ces chiffres sont en fait très significatifs si on les rapproche des pourcentages obtenus à l'aide de méthodes de génération utilisant comme modèle un logigramme. Pour des circuits de taille beaucoup plus réduite, (environ 100 portes élémentaires) un "ensemble complet de tests" permet de détecter en réalité 90 % des pannes simples.

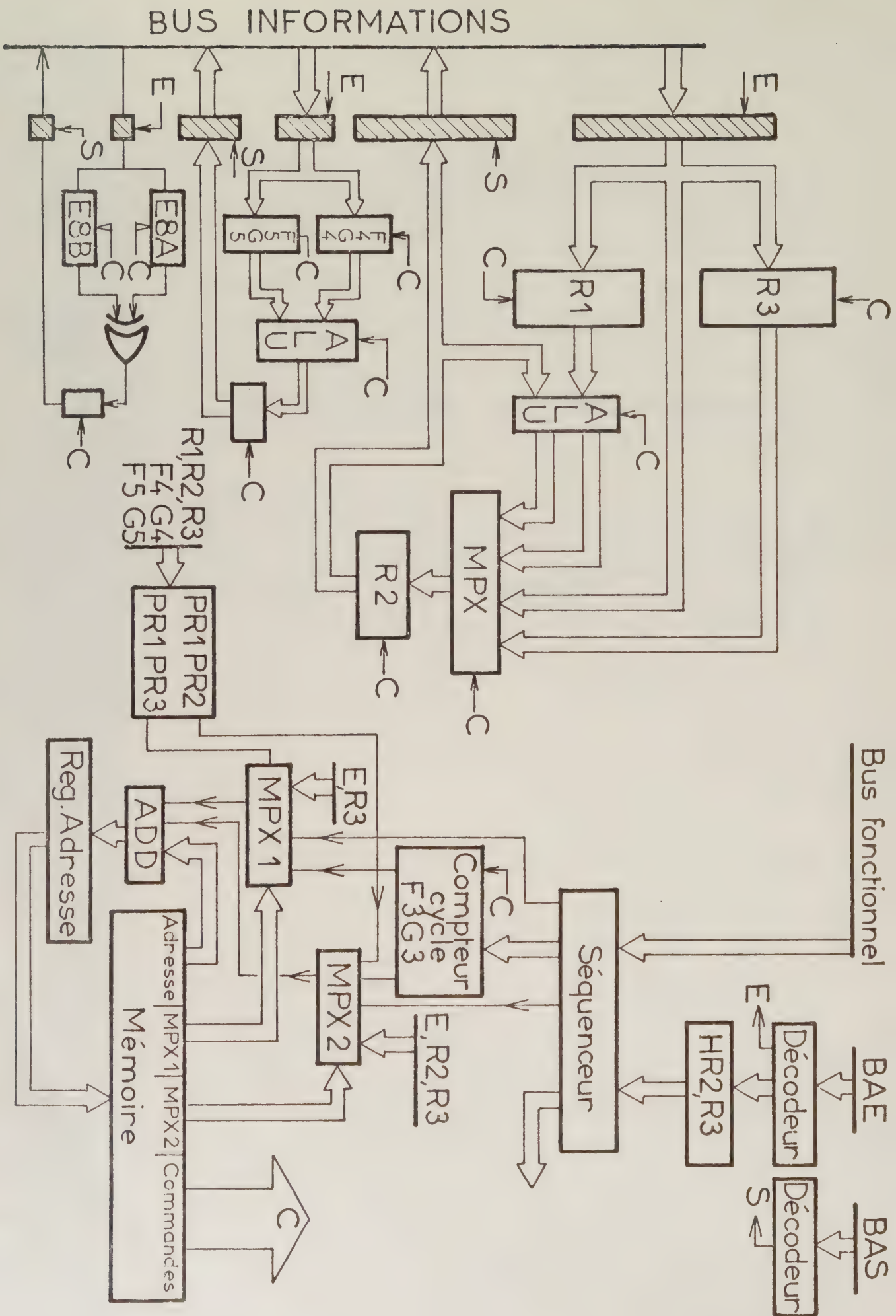


Figure 3A.1. Schéma bloc de la plaque MULTIPLICATION-DIVISION

<i>B</i> A <i>E</i> +8 <i>p</i> 0	<i>Bus adresse entrée</i>
<i>B</i> A <i>S</i> +8 <i>p</i> 0	<i>Bus adresse sortie</i>
<i>E</i> 8 <i>A</i> +0	<i>Signe mantisse 1er opérande</i>
<i>E</i> 8 <i>B</i> +0	<i>Signe mantisse 2ème opérande</i>
<i>R</i> 1+16 <i>p</i> 0	<i>Mantisse 1er opérande</i>
<i>R</i> 2+18 <i>p</i> 0	<i>Résultat mantisse, mantisse 2ème op. pour div.</i>
<i>R</i> 3+18 <i>p</i> 0	<i>Mantisse 2ème opérande pour mul.</i>
<i>F</i> 4 <i>G</i> 4+6 <i>p</i> 0	<i>Signe caract. et caract. 1er opérande</i>
<i>F</i> 5 <i>G</i> 5+6 <i>p</i> 0	<i>Signe caract. et caract. 2ème opérande</i>
<i>F</i> 3 <i>G</i> 3+5 <i>p</i> 0	<i>Compteur de cycle</i>
<i>Q</i> D7 <i>G</i> 1+7 <i>p</i> 0	<i>Signe caract. et caract. du résultat</i>
<i>P</i> R1 <i>P</i> R2+2 <i>p</i> 0	<i>Test de normalisation pour la division</i>
<i>P</i> R1 <i>P</i> R3+2 <i>p</i> 0	<i>Test de normalisation pour la multiplication</i>
<i>Y</i> R2+ <i>R</i> 2[18]	<i>Test d'arrondi pour la multiplication</i>
<i>R</i> 1 <i>R</i> 2+ <i>R</i> 3[16,17]	<i>Test d'arrondi pour la division</i>

Figure 3A.2. Liste des identificateurs
de la plaque

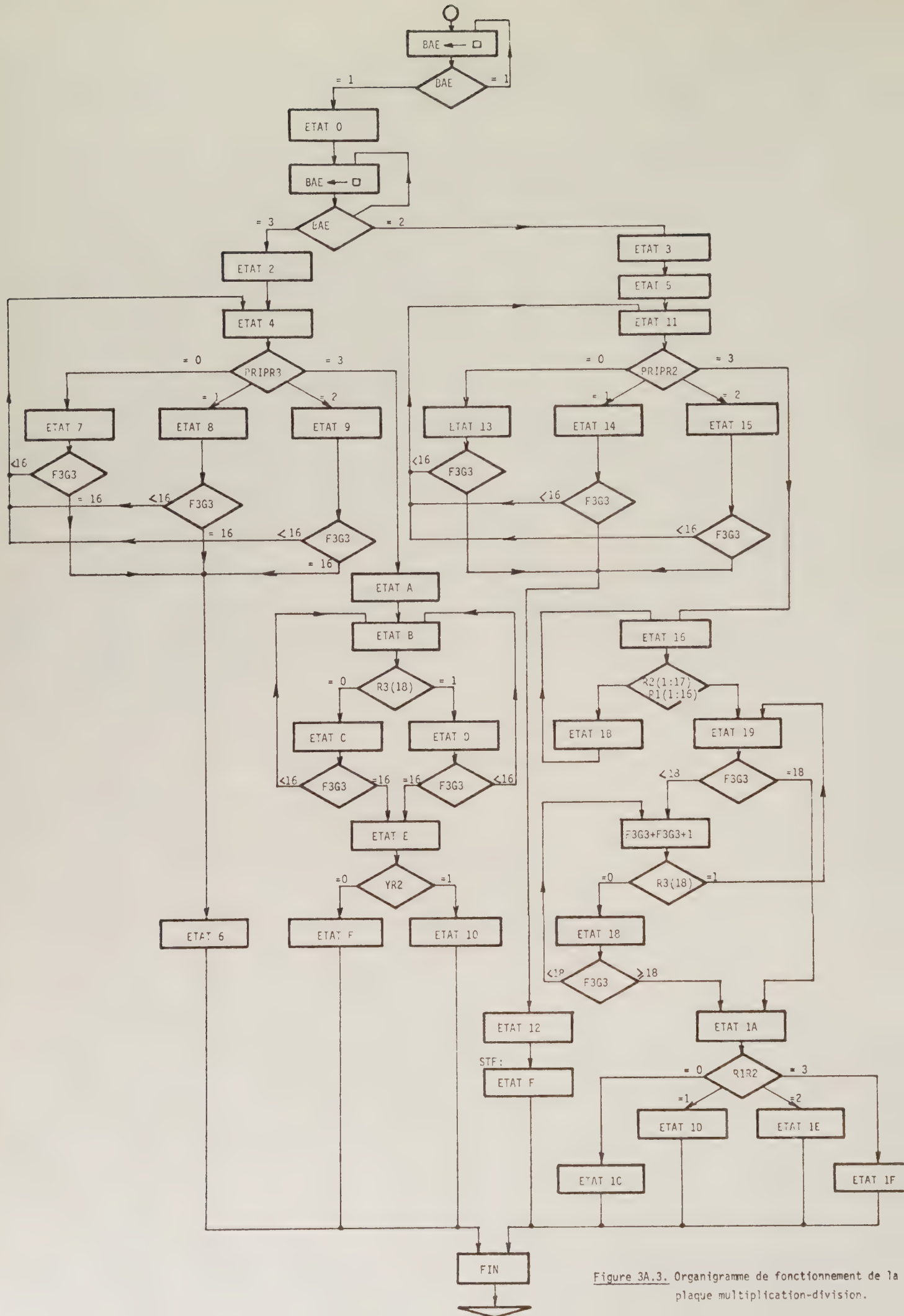


Figure 3A.3. Organigramme de fonctionnement de la plaque multiplication-division.

```

      VMULTI[[]]V
    ▽ MULTI
[1]  BAE←[]
[2]  →1+2×BAE=1
[3]  ETAT0
[4]  ETAT1
[5]  →ST2×1BAE=3
[6]  ETAT3
[7]  ETAT5
[8]  ST11:ETAT11
[9]  →((13)=PR1PR2)/ST14,ST15,ST16
[10] ST13:ETAT13
[11] →((F3G3<16),1)/ST11,ST12
[12] ST14:ETAT14
[13] →((F3G3<16),1)/ST11,ST12
[14] ST15:ETAT15
[15] →ST11×1F3G3<16
[16] ST12:ETAT12
[17] →STF
[18] ST16:RAP02
[19] ST1A:ETAT1A
[20] →((13)=P1P2)/ST1D,ST1E,ST1F
[21] ST1C:ETAT1C
[22] →STOP
[23] ST1D:ETAT1D
[24] →STOP
[25] ST1E:ETAT1E
[26] →STOP
[27] ST1F:ETAT1F
[28] →STOP
[29] ST2:ETAT2
[30] ST4:ETAT4
[31] →((13)=PR1PF3)/ST8,ST9,STA
[32] ST7:ETAT7
[33] →((F3G3=16),1)/ST6,ST4
[34] ST8:ETAT8
[35] →((F3G3=16),1)/ST6,ST4
[36] ST9:ETAT9
[37] →((F3G3=16),1)/ST6,ST4
[38] ST6:ETAT6
[39] →STOP
[40] STA:RAP01
[41] STE:ETATE
[42] →ST10×1YR2
[43] STF:ETATF
[44] →STOP
[45] ST10:ETAT10
[46] STOP:FIN
    ▽
    ▽ VRAB01[[]]V
    ▽ RAB01
[1]  STA:ETATA
[2]  STB:ETATB
[3]  →STD×1R3[18]
[4]  ETATC
[5]  →((F3G3=16),1)/0,STB
[6]  STD:ETATD
[7]  →((F3G3=16),1)/0,STB
    ▽
    ▽ RAP02[[]]V
    ▽ RAP02
[1]  ST16:ETAT16
[2]  →ST19×1((17p2)1P2[117])<(16p2)1P1
[3]  ETAT1P
[4]  →ST16
[5]  ST19:ETAT19
[6]  →0×1F3G3=18
[7]  ST17:F3G3+F3G3+1
[8]  →ST19×1R3[18]≠0
[9]  ETAT18
[10] →ST17×1F3G3<18
    ▽

```

Figure 3A.4. Description fonctionnelle de la plaque

```

      ∇ ETAT0[[]] ∇
[1] ∇ ETAT0
[2] E8A+[]
[3] R1+[]
[4] F4G4+[]
      G82+G83+F3G3+0
      ∇
      ∇ ETAT1[[]] ∇
[1] ∇ ETAT1
[2] BAE+[]
[3] +1+2×(BAE=2)∇(BAE=3)
[4] R2+18ρE71+0
      G82+G83+1
      ∇
      ∇ ETAT2[[]] ∇
[1] ∇ ETAT2
[2] E8B+[]
[3] R3+0,0,(R3+[])
      F5G5+[]
      ∇
      ∇ ETAT3[[]] ∇
[1] ∇ ETAT3
[2] E8B+[]
[3] R2+0,(R2+[]),0
      F5G5+[]
      ∇
      ∇ ETAT4[[]] ∇
[1] ∇ ETAT4
[2] F3G3+F3G3+1
[3] G82+G83+0
      PR1PR3+(R3[3]∇(∧/F5G5=1,(4ρ0),1))+2×(R1[1]∇(∧/F4G4=1,(4ρ0),1)
      ∇
      ∇ ETAT5[[]] ∇
[1] ∇ ETAT5
      F3G3+G82+G83+0
      ∇
      ∇ ETAT6[[]] ∇
[1] ∇ ETAT6
      R2+18ρQD7G1+6ρF3G3+0
      ∇
      ∇ ETAT7[[]] ∇
[1] ∇ ETAT7
[2] R1+R1[1+1(ρR1)-1],0
[3] R3+R3[1+1(ρR3)-1],0
[4] F4G4+(6ρ2)∇((6ρ2)∇F4G4)-1
      F5G5+(6ρ2)∇((6ρ2)∇F5G5)-1
      ∇
      ∇ ETAT8[[]] ∇
[1] ∇ ETAT8
[2] R1+R1[1+1(ρR1)-1],0
      F4G4+(6ρ2)∇((6ρ2)∇F4G4)-1
      ∇
      ∇ ETAT9[[]] ∇
[1] ∇ ETAT9
[2] R3+R3[1+1(ρR3)-1],0
      F5G5+(6ρ2)∇((6ρ2)∇F5G5)-1
      ∇
      ∇ ETATA[[]] ∇
[1] ∇ ETATA
      R2+18ρF3G3+0
      ∇
      ∇ ETATB[[]] ∇
[1] ∇ ETATB
[2] F3G3+F3G3+1
      XR3+R3[18]
      ∇
      ∇ ETATC[[]] ∇
[1] ∇ ETATC
[2] R3+0,R3[1(ρR3)-1]
      R2+0,R2[1(ρR2)-1]
      ∇
      ∇ ETATD[[]] ∇
[1] ∇ ETATD
[2] R3+0,R3[1(ρR3)-1]
[3] AR2+(17ρ2)∇R2[117]
[4] AR1+(16ρ2)∇R1
[5] ALU+AR1+AR2
[6] ALU+(17ρ2)∇ALU
      R2+0,ALU
      ∇

```

Figure 3A.4. (Suite)

```

    VETATE[[]]V
  ▽ ETATE
[1] F3G3+0
[2] YR2+R2[18]
[3] BF4G4+F4G4[1],F4G4[1],F4G4
[4] BF5G5+F5G5[1],F5G5[1],F5G5
[5] ALUC+(9ρ2)τ((8ρ2)⊥BF4G4)+((8ρ2)⊥BF5G5)
[6] QD7G1+ALUC[2+ι(ρALUC)-2]
  ▽
  VETATF[[]]V
  ▽ ETATF
[1] R2+R2[1+ι(ρR2)-1],0
[2] →0×ι(((~QD7G1[1])VQD7G1[2])≠0)
[3] R2+18ρQD7G1+6ρ0
  ▽
  VETAT10[[]]V
  ▽ ETAT10
[1] AR2+(17ρ2)⊥R2[ι17]
[2] ALU+AR2+1
[3] ALU+(17ρ2)τALU
[4] R2+ALU[1+ι16],0,0
[5] →0×ι×(((~QD7G1[1])VQD7G1[2])≠0)
[6] R2+18ρQD7G1+6ρ0
  ▽
  VETAT11[[]]V
  ▽ ETAT11
[1] F3G3+F3G3+1
[2] PR1PR2+(R2[2]V(∧/F5G5=1,(4ρ0),1))+2×(R1[1]V(∧/F4G4=1,(4ρ0),1))
  ▽
  VETAT12[[]]V
  ▽ ETAT12
[1] R2+18ρF3G3+0
[2] QD7G1+7ρ0
  ▽
  VETAT13[[]]V
  ▽ ETAT13
[1] R1+R1[1+ι(ρR1)-1],0
[2] R2+R2[1+ι(ρR2)-1],0
[3] F4G4+(6ρ2)τ((6ρ2)⊥F4G4)-1
[4] F5G5+(6ρ2)τ((6ρ2)⊥F5G5)-1
  ▽
  VETAT14[[]]V
  ▽ ETAT14
[1] R1+R1[1+ι(ρR1)-1],0
[2] F4G4+(6ρ2)τ((6ρ2)⊥F4G4)-1
  ▽
  VETAT15[[]]V
  ▽ ETAT15
[1] R2+R2[1+ι(ρR2)-1],0
[2] F5G5+(6ρ2)τ((6ρ2)⊥F5G5)-1
  ▽
  VETAT16[[]]V
  ▽ ETAT16
[1] F3G3+0
  ▽
  VETAT17[[]]V
  DEFN ERROR
  ▽ ETAT17[[]]V
  ▽
  VETAT18[[]]V
  ▽ ETAT18
[1] ALU+(17ρ2)τ((17ρ2)⊥R2[ι17])+(16ρ2)⊥R1
[2] R2+ALU[1+ι16],0,0
[3] R3+R3[1+ι(ρR3)-1],~ALU[1]
  ▽
  VETAT19[[]]V
  ▽ ETAT19
[1] ALU+(17ρ2)τ((17ρ2)⊥R2[ι17])-(16ρ2)⊥R1
[2] R2+ALU[1+ι16],0,0
[3] R3+R3[1+ι(ρR3)-1],~ALU[1]
  ▽
  VETAT1A[[]]V
  ▽ ETAT1A
[1] R2+0,R3[ι16],0
[2] R1R2+2×R3[16]+R3[17]
  ▽

```

Figure 3A.4. (Suite)

```

      VETAT1B[[]]V
    ▽ ETAT1B
  [1] R2+0,R2[1+(pR2)-1]
  [2] F5G5+(6p2)T((6p2)⊥F5G5)+1
    ▽
      VETAT1C[[]]V
    ▽ ETAT1C
  [1] AR2+(17p2)⊥R2[117]
  [2] ALU+AR2-1
  [3] ALU+(17p2)TALU
  [4] R2+ALU[1+116],0,0
  [5] BF4G4+F4G4[1],F4G4[1],F4G4
  [6] BF5G5+F5G5[1],F5G5[1],F5G5
  [7] ALUC+(9p2)T((8p2)⊥BF5G5)-((8p2)⊥BF4G4)
  [8] QD7G1+ALUC[2+1(pALUC)-2]
  [9] →0×1×(((~QD7G1[1])∨QD7G1[2])≠0)
  [10] R2+18p0
  [11] QD7G1+7p0
    ▽
      VETAT1D[[]]V
    ▽ ETAT1D
  [1] R2+R2[1+1(pR2)-1],0
  [2] BF4G4+F4G4[1],F4G4[1],F4G4
  [3] BF5G5+F5G5[1],F5G5[1],F5G5
  [4] ALUC+(9p2)T((8p2)⊥BF5G5)-((8p2)⊥BF4G4)
  [5] QD7G1+ALUC[2+1(pALUC)-2]
  [6] →0×1×(((~QD7G1[1])∨QD7G1[2])≠0)
  [7] R2+18pQD7G1+7p0
    ▽
      VETAT1E[[]]V
    ▽ ETAT1E
  [1] R2+R2[1+1(pR2)-1],0
  [2] BF4G4+F4G4[1],F4G4[1],F4G4
  [3] BF5G5+F5G5[1],F5G5[1],F5G5
  [4] ALUC+(9p2)T((8p2)⊥BF5G5)-((8p2)⊥BF4G4)
  [5] QD7G1+ALUC[2+1(pALUC)-2]
  [6] →0×1×(((~QD7G1[1])∨QD7G1[2])≠0)
  [7] R2+18pQD7G1+7p0
    ▽
      VETAT1F[[]]V
    ▽ ETAT1F
  [1] AR2+(17p2)⊥R2[117]
  [2] ALU+AR2+1
  [3] ALU+(17p2)TALU
  [4] R2+ALU[1+116],0,0
  [5] BF4G4+F4G4[1],F4G4[1],F4G4
  [6] BF5G5+F5G5[1],F5G5[1],F5G5
  [7] ALUC+(9p2)T((8p2)⊥BF5G5)-((8p2)⊥BF4G4)
  [8] QD7G1+ALUC[2+1(pALUC)-2]
  [9] →0×1×(((~QD7G1[1])∨QD7G1[2])≠0)
  [10] R2+18pQD7G1+7p0
    ▽
      VFIN[[]]V
    ▽ FIN
  [1] BAS+[]
  [2] →1+2×(BAS=2)
  [3] []+E8A≠E8B
  [4] []←R2
  [5] []←QD7G1[1+1(pQD7G1)-1]
    ▽

```

Figure 3A.4. (Suite)

$\nabla \text{ } RR \leftarrow DG \text{ } R$ décalage gauche de R; entree=0
 $RR \leftarrow R[1+i(pR)-1], 0 \nabla$

$\nabla \text{ } RR \leftarrow DD \text{ } R$ décalage droit de R: entree=0
 $RR \leftarrow 0, R(i[pR]-1) \nabla$

$\nabla \text{ } RR \leftarrow X \text{ } DG \text{ } R$ décalage gauche de R: entrée=X
 $RR \leftarrow R[1+i(pR)-1], X \nabla$

$\nabla \text{ } RR \leftarrow X \text{ } DD \text{ } R$ décalage droit de R:entrée=X
 $RR \leftarrow X, R(i[pR]-1) \nabla$

$\nabla \text{ } B \leftarrow CPT \text{ } A$ incrémentation de A
 $B \leftarrow ((pA) \downarrow p2) \uparrow ((pA) \downarrow p2) \downarrow A \uparrow +1 \nabla$

$\nabla \text{ } B \leftarrow DECPT \text{ } A$ décrémentation de A
 $B \leftarrow ((pA) \downarrow p2) \uparrow ((pA) \downarrow p2) \downarrow A -1 \nabla$

$\nabla \text{ } RR \leftarrow RA \text{ } MOINS \text{ } RB$ différence de deux nombres binaires
 $RR \leftarrow ((1+(pRB[(pRA)] \downarrow p2) \uparrow ((pRA) \downarrow p2) \downarrow RA) - ((pRB) \downarrow p2 \downarrow RB) \nabla$

$\nabla \text{ } RR \leftarrow RA \text{ } PLUS \text{ } RB$ somme de deux nombres binaires
 $RR \leftarrow ((1+(pRB[(pA)] \downarrow p2) \uparrow ((pRA) \downarrow p2) \downarrow RA) + ((pRB) \downarrow p2 \downarrow RB) \nabla$

Figure 3A.5. Définition des opérateurs
spécifiques de la plaque.

CONCLUSION

La vérification d'un système logique consiste à s'assurer que le système réel réalise bien un modèle mathématique M_0 qui décrit son comportement. M_0 est un ensemble de propriétés fonctionnelles élémentaires. Pour les systèmes discrets, il s'agit d'un ensemble dénombrable.

A partir d'un certain niveau de complexité la vérification exhaustive de ces propriétés est pratiquement impossible. On définit alors un modèle de panne pour réduire le nombre des propriétés à vérifier. Si P est l'ensemble des pannes déduites de ce modèle, il faut à priori que la probabilité de $\bar{P}$ (complémentaire de P par rapport à l'ensemble U de toutes les pannes possibles) soit inférieure à un ϵ donné.

L'ensemble P induit un ensemble M_P de modèles de fonctionnement. Le problème de la détection est alors la recherche d'un sous-ensemble de propriétés de M_0 qu'il est suffisant de vérifier pour distinguer M_0 de l'ensemble M_P .

Deux approches sont alors possibles :

La première consiste à identifier M_0 dans l'ensemble $M_0 \cup M_P$. Cette solution a été développée en particulier dans la théorie des automates.

La seconde est possible si le modèle de panne est tel que tout élément de M_P est comparable à M_0 . C'est cette dernière solution que nous avons développée.

Elle consiste à considérer pour une panne p donnée le modèle M_p et à le comparer à M_0 pour en déduire une propriété permettant de les séparer. Les méthodes qui en découlent, dépendent des modèles de fonctionnement et de panne considérés.

Ainsi, il est fondamental de définir l'ensemble des pannes de façon qu'il soit possible, à partir du modèle de fonctionnement M_0 , de déduire un modèle M_p de même nature que M_0 .

Cette remarque conduit à la définition des collages de connexions du système logique à vérifier. Ce modèle de panne ne satisfait pas forcément la condition probabilité de $\bar{P}$ inférieure à un ϵ donné. Il semble alors qu'il y ait conflit entre ces deux conditions. En fait, il a été montré, à propos de systèmes logiques, que les propriétés permettant de distinguer M_0 de l'ensemble M_P séparaient M_0 d'un sous-ensemble U' de U bien plus grand que M_P . Malheureusement, U' est spécifique de l'ensemble des propriétés choisies et l'on ne peut avoir une mesure exacte qu'après les avoir choisies.

La démarche employée est donc d'admettre un modèle de panne à priori suffisamment simple à traiter puis de le valider à posteriori en vérifiant que la probabilité de $\bar{U}$ est bien inférieure à la valeur ϵ choisie.

Pour les systèmes combinatoires, le problème reste assez simple. D'une manière générale on peut avoir soit un modèle purement fonctionnel, par exemple algébrique, soit un modèle de structure de réalisation, par exemple logigramme. Les difficultés de passer de l'un à l'autre montrent qu'il est, dans la mesure du possible, plus efficace d'employer le modèle à disposition.

Aux chapitres 1 et 2, nous avons traité, à titre d'exemple, des types de procédure que l'on peut mettre en oeuvre sur un modèle algébrique ou des modèles en logigramme.

L'avantage des solutions algébriques réside dans leur généralité. Certaines sont indépendantes du type de réalisation et conduisent à un ensemble optimal de détection, d'autres permettent de traiter le cas des collages multiples. Malheureusement, elles restent limitées à des circuits de taille faible sauf dans le cas où l'on possède déjà les fonctions algébriques associées aux connexions à tester.

Le besoin de traiter des systèmes toujours plus complexes pour lesquels on possède rarement les fonctions booléennes associées, nous a amené à étudier des solutions s'appliquant directement sur la structure de réalisation. On aboutit en général à un ensemble réduit de propriétés à vérifier dont on ne peut garantir l'optimum, mais cela n'est pas très grave en pratique.

L'inconvénient majeur de ces méthodes est qu'elles sont toujours liées au type de réalisation. Il semble donc nécessaire de concevoir, dans un d'efficacité, des solutions spécifiques des modèles de fonctionnement et de panne. Ainsi, au niveau élémentaire, il n'existe pas, à notre avis, de solution universelle mais un ensemble de solutions parmi lesquelles, en fonction des informations et des moyens disponibles, il faut savoir choisir celle qui correspond le mieux au problème à traiter.

Il semble, de plus, impossible d'éviter cette étape car on aboutirait à des solutions de type "boîte noire", c'est-à-dire ne prenant pas en compte la structure de réalisation ; donc à des ensembles de tests impossibles à appliquer pratiquement.

Bien que des méthodes de recherche directe sur une structure de réalisation soient les plus efficaces, elles restent pour une grande part à base d'énumération. En effet, une étape obligée s'apparente au problème de satisfiabilité pour lequel le temps de calcul croît très rapidement avec la taille de l'exemple à traiter. C'est à notre avis, l'une des raisons pour écarter la solution proposée pour le cas des systèmes séquentiels et qui consiste à se ramener au cas combinatoire en construisant un modèle itératif. En effet, chaque cellule de ce modèle correspond à une transition du circuit séquentiel. Il comprend donc un nombre de cellules égales à la longueur de la séquence recherchée. Par exemple, pour un circuit séquentiel de 1000 portes élémentaires, le calcul d'une séquence de longueur 5 conduit à un circuit combinatoire de 5000 portes.

A cet argument viennent s'ajouter d'autres inconvénients :

- la nécessité de localiser les variables internes ce qui, en dehors du calcul nécessaire, implique une hypothèse sur la localisation des retards. Cette supposition qui peut avoir un sens pour un circuit sans panne est très douteuse dans le cas contraire et nécessite toujours une validation des solutions obtenues. On constate d'ailleurs un pourcentage de rejet croissant très rapidement avec la taille des systèmes traités ;

- l'impossibilité, due à la taille du problème, de prendre en compte au niveau de la génération, les retards de chaque élément amène à conclure que la solution pratique ne peut qu'être heuristique. Encore faut-il au moins utiliser, quand cela s'avère possible, le modèle simplifié à l'intérieur de son domaine de validité ;

- la difficulté de traduire les contraintes au niveau d'une structure de réalisation aboutit à des séquences de vérification n'appartenant pas au fonctionnement normal.

Ces différents points trouvent une solution si l'on tient compte, au niveau de la génération des tests, du fonctionnement normal du système. En effet, il correspond au domaine de validité du modèle, il contient bien tous les tests, il induit une partition du système qui est celle utilisée au cours de la conception de tout système complexe. Nous avons donc développé une description s'appuyant sur deux éléments :

- .L'un est la description fonctionnelle sous forme d'un organigramme dont les instructions correspondent à l'ensemble des modifications ordonnées que subissent les variables d'état, les entrées et les sorties ;

- .L'autre correspond à une description de la structure non nécessairement homogène et dont la seule contrainte est que tout sous-ensemble attaché à une instruction corresponde à un seul type de réalisation.

Ces deux éléments sont liés par une application qui peut être implicitement réalisée par une identité des noms affectés aux identificateurs.

Les instructions de la partie fonctionnelle sont écrites dans un langage évolué (A.P.L.) qui permet de traduire les modifications et les commandes usuelles à l'aide de primitives du langage.

La dualité fonction-structure permet, grâce à la partie fonction, la compréhension humaine et donc le choix d'un chemin dans l'organigramme. La structure rend possible l'automatisation des tâches sujettes à erreurs comme le choix d'une image réciproque, la recherche d'un test au niveau d'une instruction, et la vérification de la mise en évidence de la panne.

Nous avons montré la nécessité d'une nouvelle description des systèmes logiques complexes. Le fait important est qu'elle regroupe deux types d'information, Mission-Réalisation, qui existaient par ailleurs mais étaient utilisés séparément.

Notre conviction est que de nombreux problèmes d'analyse et de synthèse des systèmes logiques complexes doivent être repris avec cette nouvelle description. Bien sûr, la plupart de ces problèmes ont reçu des solutions s'appuyant simplement sur la réalisation, mais leur application est souvent limitée à des petits systèmes. Ainsi à la suite de E.W. DJISTRA nous affirmerons l'impossibilité fondamentale de comparer deux choses quelconques qui, d'une certaine façon, diffèrent d'un facteur d'au moins cent.

Parce que l'outil utilisé conditionne le produit, l'importance du langage utilisé est incontestable. Nous avons choisi un langage série parce que l'ordre augmente la clarté et donc la compréhension. Mais il existe d'autres paramètres tout aussi importants :

Tout d'abord la clarté a bien évidemment un aspect quantitatif. Aussi pour des systèmes à évolution essentiellement parallèle, il semblerait

souhaitable d'ajouter certaines propriétés au langage pour éviter d'aboutir à une description trop volumineuse.

Enfin, la structure même du programme est un fait important. Il ne concerne pas uniquement le problème de détection mais aussi celui de la vérification d'une conception et la manière de concevoir un système discret.

Ces notions apparaissent par exemple comme une discipline de programmation (DJI 1) dont le mot d'ordre est : plutôt que de rechercher la dernière erreur d'éviter de trouver la première (MIL 1).

La définition des règles précises à respecter au niveau de la structure du programme doit permettre de produire des réalisations plus simples à vérifier.

Enfin, les liens étroits qui existent entre la vérification du matériel et celle du logiciel, qui à l'heure actuelle, est loin d'être résolue, nous autorise à penser que les résultats de cette étude pourront être généralisés à ce domaine.

BIBLIOGRAPHIE

- (AKE 1) S.B. AKERS, "On a Theory of Boolean functions",
J. Soc. Ind. Math, Vol.7, n°4, Décembre 1959.

- (ARM 1) D.B. ARMSTRONG, "A deductive method for simulating faults in logic
circuits",
IEEE T.C., Mai 1972, pp.464/471.

- (AZE 1) P. AZEMA, M. COURVOISIER, J.P. RICHARD, "Méthode de détection des
pannes dans les systèmes séquentiels", Congrès IFAC, Paris, Juin 1972.

- (BAR 1) M.R. BARBACCI, "A comparison of Register Transfert Languages for
descri ing Computers and Digital Systems",
Report of Depart. of Comp. Sc., Carnegie- Mellon University, Mars 1973.

- (BEN 1) R.G. BENNETTS, "Computer Aided generation of Test Sequences for
logical systems - Part 1 and 2",
Ph.D. Thesis, University of Southampton, 1973.

- (BEN 2) R.G. BENNETTS, "A realistic approach to detection test set generation
for combinational logic circuits",
B S C Computer Journal, Vol.15, n°2, 1972, pp.238/246.

- (BRE 1) M.A. BREUER, L. HARRISON, "Procedures for eliminating static and
dynamic hazards in test generation",
IEEE Group Repository report R 73-251, 28 Mars 1973.

- (CAN 1) R.H. CANADAY, "Two dimentional iterative logic",
Proc. Fall Joint Comp. Conf. , 1965, pp.343/358.

- (CAR 1) W.C. CARTER, "Experiments in proving design correctness for micro-
program controlled computers",
Symposium on F.T.C., Illinois, pp.5-22/5-25, 1974.

- (CAV 1) J.R. CAVALCANTI FILHO, "Génération de tests de localisation des pannes dans les circuits digitaux complexes",
Thèse P.U.C. Dépt. Génie Electrique, Février 1974.
- (CHI 1) C. CHICOIX, J. ERCEAU, F. BLANCA, P. PRECHOUX, "Dialog. Programme d'élaboration des séquences de tests de détection dans les multiples combinatoires",
Coll. Int. sur la Microélectronique avancée, Paris, Avril 1970.
- (CHA 1) D.K. CHIA, M.Y. HSIAO, "A homing sequence generation algorithm for fault detection in asynchronons digital circuits",
Int. Symp. on F.T.C., Newton, Massachussetts, Juin 1972, pp.137/142.
- (CHI 2) C. CHICOIX, C. DURANTE, J. ERCEAU, J.P. RICHARD, "Influence de la topologie sur la détection et la localisation des pannes dans les sous-ensembles logiques",
Symp. Int. Sur les Systèmes Logiques, Bruxelles 1969.
- (COO 1) S. COOK, "The complexity of theorem proving procedures",
Third ACM Symp. on Theory of Computing, pp.151/158, 1970.
- (COU 1) M. COURVOISIER, "Etude des systèmes logiques de commande asynchrones à évolutions simultanées",
Thèse d'état, Toulouse, 26 Février 1974.
- (DAV 1) M. DAVIO, P. PIRET, "Les dérivées booléennes et leur application au diagnostic",
Symp. Int. sur les Systèmes logiques, Bruxelles 1969.
- (DEC 1) P. DECHIZEAUX, "Application à l'algèbre de Boole de la notion de dérivée",
Séminaire de logique IMAG, Février 1965.

- (DIA 1) M. DIAZ, J.P. RICHARD, M. COURVOISIER, "A note on minimal and quasi minimal essential sets in complex directed graphs",
IEEE T. on C.T., vol. CT 19, n°5, 1972.
- (DIA 2) M. DIAZ, M. COURVOISIER, J.P. RICHARD, "Une approche à la détection en ligne des pannes dans les systèmes séquentiels synchrones",
Colloque CEMAL, Septembre 1972.
- (DIJ 1) E.W. DIJKSTRA, O.J. DAHL, C.A.R. HOARE, "Notes on structured programming", Academic Press, 1972.
- (DOU 1) J.E. DOUCET, G. GELIS, "Programme de test pour les pannes multiples"
Note interne L.A.A.S., S.L.N. 73.I.9.
- (DUR 1) C. DURANTE, J. ERCEAU, J.P. RICHARD, "Sur la détection des pannes dans les ensembles logiques",
Journées d'Electronique de Toulouse, 1968.
- (ERC 1) J. ERCEAU, "Contribution à l'étude de la notion de différence booléenne et à ses applications au diagnostic logique",
Thèse d'Etat, Toulouse, Février 1973.
- (FAD 1) A. FADINI, "Operatori che estando alle algebre di boole la nozione di derivata",
Math. Ital. n°89, pp.43/64, 1961.
- (FAL 1) A.D. FALKOFF, K.E. IVERSON, E.H. SUSSENGUTH, "Formal description of system 360",
I.B.M. Syst. J., Vol.3, pp.198/262, 1964.
- (FRI 1) A.D. FRIEDMAN, P.R. MENON, "Fault detection in digital circuits",
Prentice Hall, 1971.

- (GAR 1) M.R. GAREY, JOHNSON, STOCKMEYER, "Some simplified N-P complete problems",
ACMSIGACT, Proc. 1974.

- (GIA 1) R. GIANFRANCO, S. PUTZOLU, J.P. ROTH, "An heuristic algorithm for the testing of asynchronous circuits",
I.E.E.E. T.C., Vol. C.20, n°6, Juin 1971, pp.639/647.

- (GON 1) G. GONENG, "A method for the design of fault detection experiments",
I.E.E.E. T.C., Juin 1970, pp.551/558.

- (IPP 1) J.C. IPPOLITO, "Contribution à l'implantation de fonctions logiques sur des réseaux cellulaires",
Thèse d'Etat, U.P.S. Toulouse, Décembre 1971.

- (IPP 2) J.C. IPPOLITO, J.P. RICHARD, "Circuit cellulaire facilement testable"
Colloque CEMAL, Toulouse, 1972.

- (IVE 1) K.E. IVERSON, A.D. FALKOFF, "A.P.L. Manuel de référence",
(Traduction française, Imprimerie I.B.M. FRANCE, 1970).

- (KUB 1) H. KUBO, "A procedure for generating test sequences to detect sequential circuits failures",
NEC Research Dev., Octobre 1968.

- (LAP 1) J.C. LAPRIE, A. COSTES, C. LANDRAULT, A.L. BOGADO, "Projet ASMODEE",
Journées d'Etude sur les calculateurs embarqués et leurs applications, Toulouse, Juin 1974.

- (MAR 1) P.N. MARINOS, "A method of deriving minimal complete sets of test input sequences using boolean differences",
Proceeding of the 1970 I.E.E.E., Int. Group. Conf. Juin 1970.

- (MIL 1) H.D. MILLS, "On the development of large reliable programs",
I.E.E.E. Symposium on Computer Software Reliability, N.Y., May 1973.

- (MIN 1) R.C. MINNICK, "Cellular linear input logic",
AD 443-802, Février 1974.
- (MIN 2) R.C. MINNICK, "Survey of microcellular research",
J. A. C. M., April 1967, pp.203/241.
- (MUK 1) A. MUKHOPADHYAY, "Recent developments in switching theory",
Academics Press, N.Y., 1971.
- (PRI 1) A.C. PRIOR, "An extension of boolean differentiation to define the
test for a specific logic fault in combinational logic network",
A paraître.
- (RIC 1) J.P. RICHARD, C. LANDRAULT, "Description fonctionnelle, application
à la plaque multiplication-division du prototype ASMODEE 01",
Note interne L.A.A.S./S.L.N., 74.I.16, 1974.
- (RIC 2) J.P. RICHARD, "Séquence de positionnement",
Note Interne L.A.A.S./S.L.N. 73.I.13, 1973.
- (RIC 3) J.P. RICHARD, "Description en vue du test",
Journée prévention des pannes dans les systèmes logiques, I.R.I.A.
4 Avril 1974.
- (RIC 4) J.P. RICHARD, "Comparaison des méthodes classiques dans l'optique
d'une caractérisation de Macro-blocs",
Rapport partiel contrat D.R.M.E.-S.E.S.A.-L.A.A.S., Publication
L.A.A.S. n°1080, Novembre 1973.
- (RIC 5) J.P. RICHARD, "Contribution à la détection des pannes dans les
circuits logiques",
Thèse de Docteur-Ingénieur, U.P.S. Toulouse, Septembre 1971.

- (RIC 6) J.P. RICHARD, M. COURVOISIER, M. DIAZ, P. AZEMA, "Detection of multiple faults in combinational circuits",
Symp. of F.T.C., Newton, 1972.
- (RIC 7) J.P. RICHARD, P. AZEMA, J.C. IPPOLITO, "Detection of multiple faults in monotonic network - Synthesis of easily testable circuits"
Int. Sympo. on F.T.C., 1973.
- (RIC 8) J.P. RICHARD, "Génération des tests de détection dans les circuits combinatoires"
Automatisme, tome XVII, n°10, Octobre 1972, pp.305/317.
- (RIC 9) J.P. RICHARD, "Fonctions booléennes et test des pannes multiples"
Séminaire I.M.A.G., Mai 1972.
- (ROT 1) J.P. ROTH, "Communication orale",
C.E.M.A.L., Toulouse, Septembre 1972.
- (SCH 1) D.R. SCHERTZ, "On the representation of digital faults",
C.F.S.T.I. Report, A.D 68-883b, Mai 1969.
- (SCH 2) D.R. SCHERTZ, G. METZE, "A new representation for faults in combinational digital circuits",
I.E.E.E. Comp. Group. Repository paper R 70-140.
- (SEL 1) F.F. SELLERS, M.Y. HSIAO, L.W. BEARNSON, "Analysis errors with the boolean difference",
I.E.E.E. T.C., Vol. C.17, Juillet 1969, pp.676/683.
- (SOU 1) M. SOUNY, J.C. LAPRIE, "Conception et réalisation d'un opérateur multiplication-division pour le microprocesseur ASM0DEE",
Note Technique L.A.A.S.:T.N.P.E., Janvier 1973.

- (TAB 1) J.H. TABARLY, "Contribution à la synthèse de tests complets de circuits digitaux par observation commande. Programme Dédale",
Thèse de Spécialité E.E.A., Juillet 1974.
- (VER 1) A. VERDILLON, "Pannes dans les réseaux acycliques"
Thèse de 3ème Cycle, Université de Grenoble, 1972.

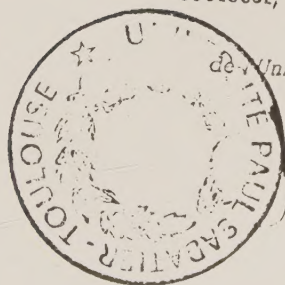
TABLE DES MATIÈRES

INTRODUCTION	3
1. Méthodes algébriques de génération des tests	11
1.1. Méthode de détection des pannes simples à partir de la différence booléenne	14
1.1.1. Définition de la différence booléenne	15
1.1.2. Condition nécessaire et suffisante	15
1.1.3. Détermination d'un ensemble optimal de tests	17
1.1.4. Détermination d'un ensemble sous-optimal de tests	17
1.1.5. Exemple d'application	18
1.2. Détection des pannes multiples	19
1.2.1. Détection des pannes multiples sur les entrées primaires	20
1.2.2. Détection des pannes multiples sur un circuit en arbre	22
1.2.3. Détection des pannes multiples dans une réalisation monotone	24
1.2.4. Détection des pannes multiples pour une réalisation quelconque	27
1.2.5. Définition d'une fonction-test	29
1.2.6. Exemples d'applications	34
1.2.7. Algorithmes mis en oeuvre	37
1.3. Conclusions	40

2. Méthodes d'obtention des tests de détection sur un logigramme	43
2.1. Méthode de transformation de logigramme	46
2.1.1. Théorie de la méthode	46
2.1.2. Solution programmée	54
2.1.3. Exemple d'application	59
2.2. Méthode graphique sur un réseau cellulaire	61
2.2.1. Structure et cellule de base du réseau	61
2.2.2. Détection des pannes dans ce type de réseau	63
a. Propriétés de transfert des cellules	63
b. Toutes les entrées de types δ sont indépendantes	64
b_1 . Génération des tests pour le collage d'une connexion	64
b_2 . Génération d'un ensemble complet de tests	67
c. Certaines entrées δ peuvent être connectées	75
2.3. Conclusions	76
3. Description Mission-Réalisation. Utilisation pour le test	79
3.1. Les défauts des modèles classiques. Propriétés du nouveau modèle	81
3.1.1. Défauts des modèles classiques	81
3.1.2. Propriétés pour un nouveau modèle	87
3.2. Description Mission-Réalisation d'un système logique	89
3.2.1. Définition de la partie Mission	89
3.2.2. Identificateurs de la description. Blocs fonctionnels	90

3.2.3. Opérateurs de la description	92
3.2.4. Evolution parallèle et description série	96
3.2.5. Description d'un sous-système	99
3.2.6. Mise en séquence de modifications	100
3.2.7. Liens avec la partie Structure de Réalisation	101
3.3. Définition formelle d'une description. Mission-Réalisation	102
3.4. Exemple d'illustration	106
3.4.1. Description de la réalisation	106
3.4.2. Description de la mission	107
3.4.3. Mise en évidence de la correspondance Mission-Réalisation	113
3.5. Utilisation de la description pour le test	115
3.5.1. Modèle de panne	115
3.5.2. Ensemble de conditions nécessaires	117
3.5.3. Conception d'un ensemble de tests	119
3.6. Conclusions	123
3A. Annexe : Exemple d'application ; La plaque multiplication-division	127
3A.1. Description Mission-Réalisation de la plaque	127
3A.2. Recherche des tests	130
C. Conclusion	141
B. Bibliographie	149

Autorisation de Soutenance accordée
TOULOUSE, le 26 NOV. 1974



Le Président
de l'Université PAUL-SABATIER :


L. LARENG

